ファミリー ベーシック™

おもしろゲーム大作戦

「マリオの走り幅とび」などゲーム19本と楽しいベーシック遊び●カセットテープ



ファミリー ベーシック™
おもしろゲーム大作戦





COMPUTER
J
YOU
3
5
2
MONEY: 11 BET: 5
1:HIT 2:STAND



*** PENGUIN'S BAR ***
SCORE
20
BONUS
1920
CLASS
3
: 5
: 3
LEFT
RIGHT
BAR

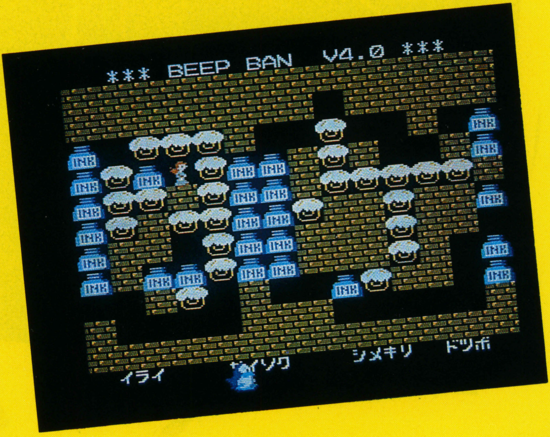


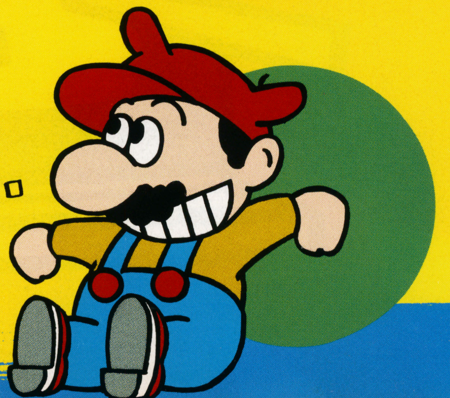


目次

おもしろゲーム大作戦!

☆ ファミリーベーシックは任天堂の商標です。





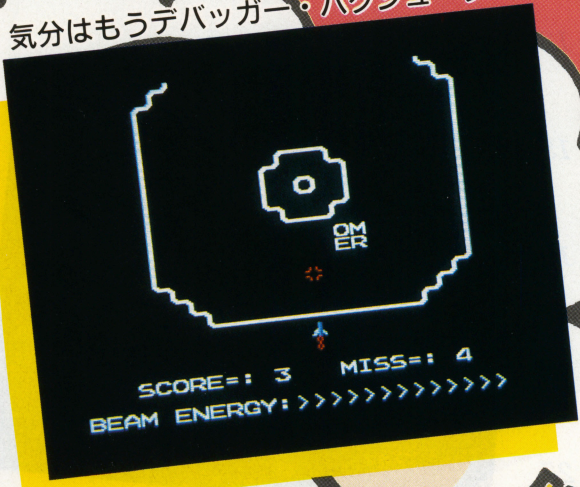
ファミリーベーシックタイプ特訓ゲーム



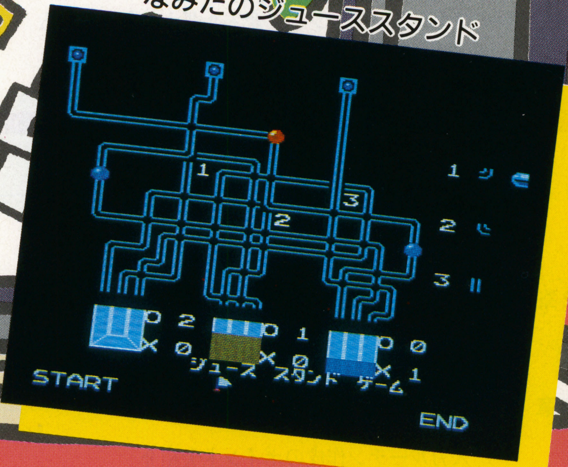
ガンバレお姉さん!! 炎のビープ番



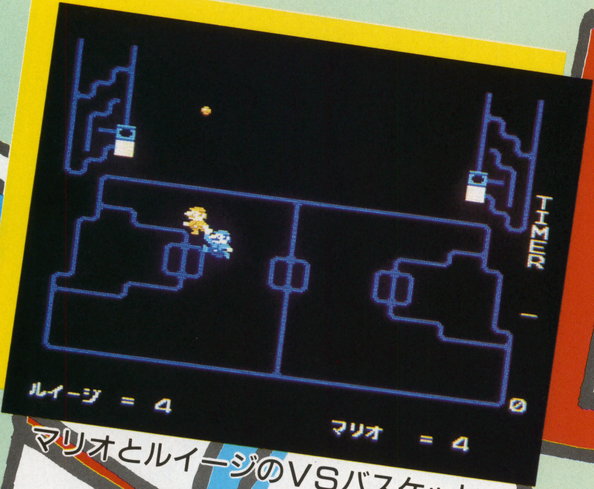
気分はもうデバッガー・バグシューター



なみだのジューススタンド



おもしろゲーム大紹介



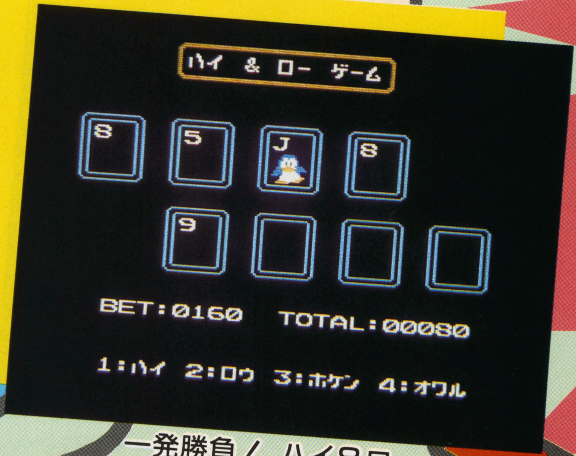
マリオとルイージのVSバスケット



21(ファミコン風ブラックジャック)



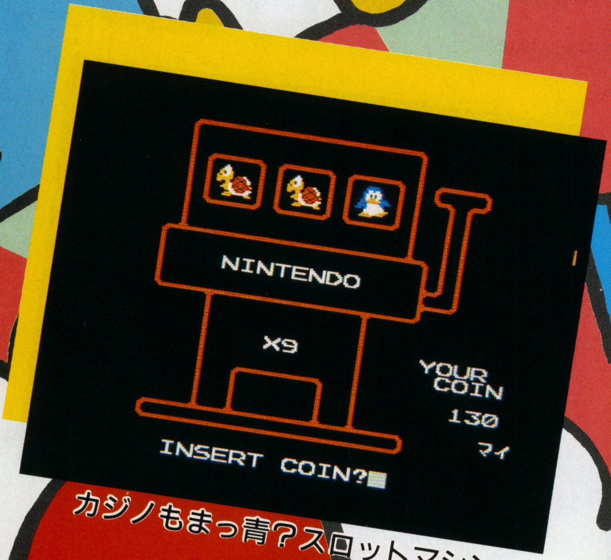
ルイージの三段とび



一発勝負！ ハイ&ロー



マリオの走り幅とび



カジノもまっ青？スロットマシン

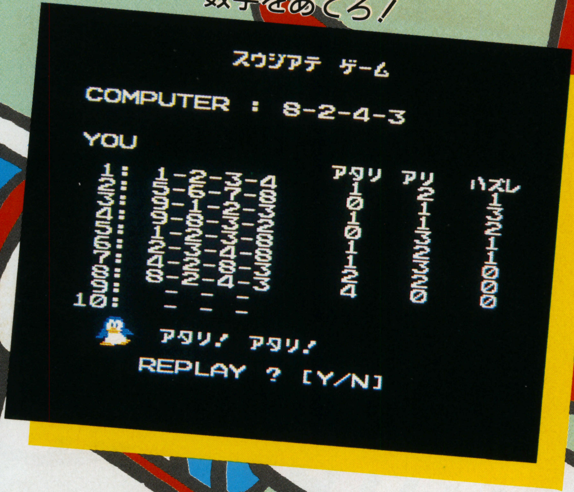


レディの走り高とび



シューティングゲーム

数字をあてろ!



ファミリーおもちゃ箱

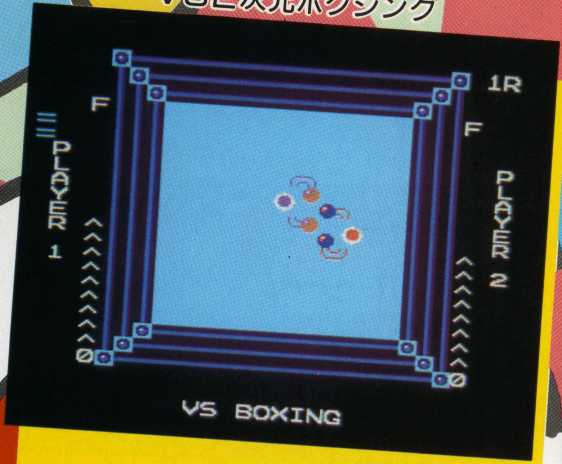


ペンジンのペンギンズバー





かっとびモトクロッサー



VS2次元ボクシング



むかしなつかしVSテレビピンポン

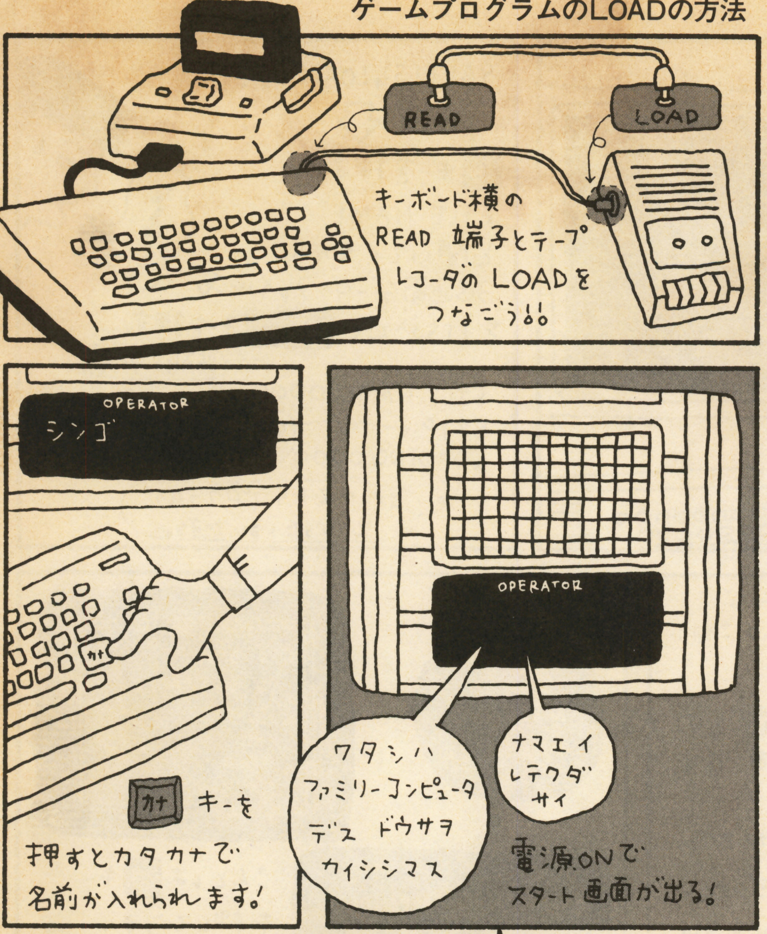


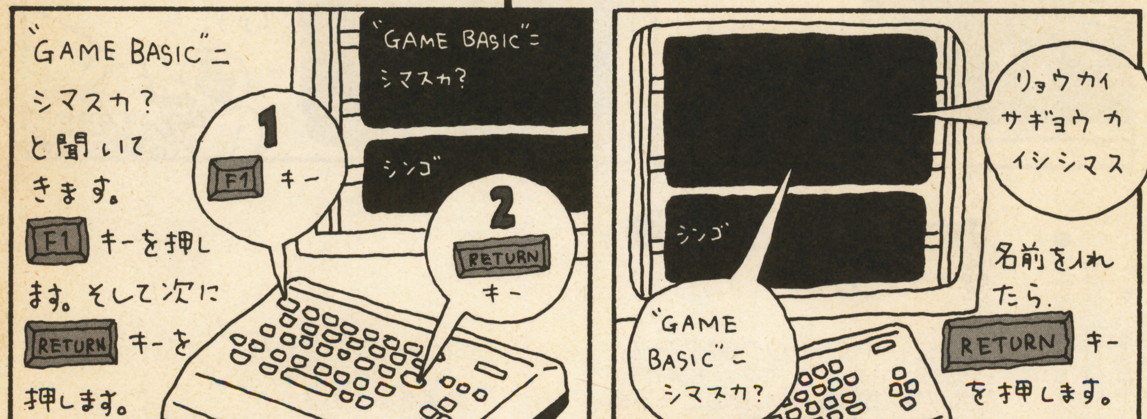
高速シューティング・スペースゲーム

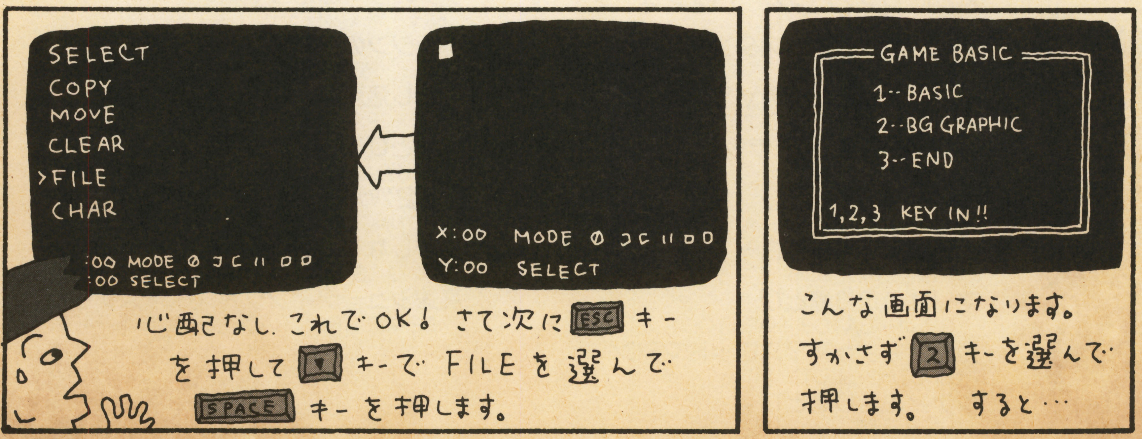
ゲームプログラムのLOADの方法

次々に飛び出すおもしろゲーム。たくさんの実践例をとり入れたテクニック解説で、あなたは知らない間におもしろゲーム人間となる。さあはじめよう!ファミリーベーシック

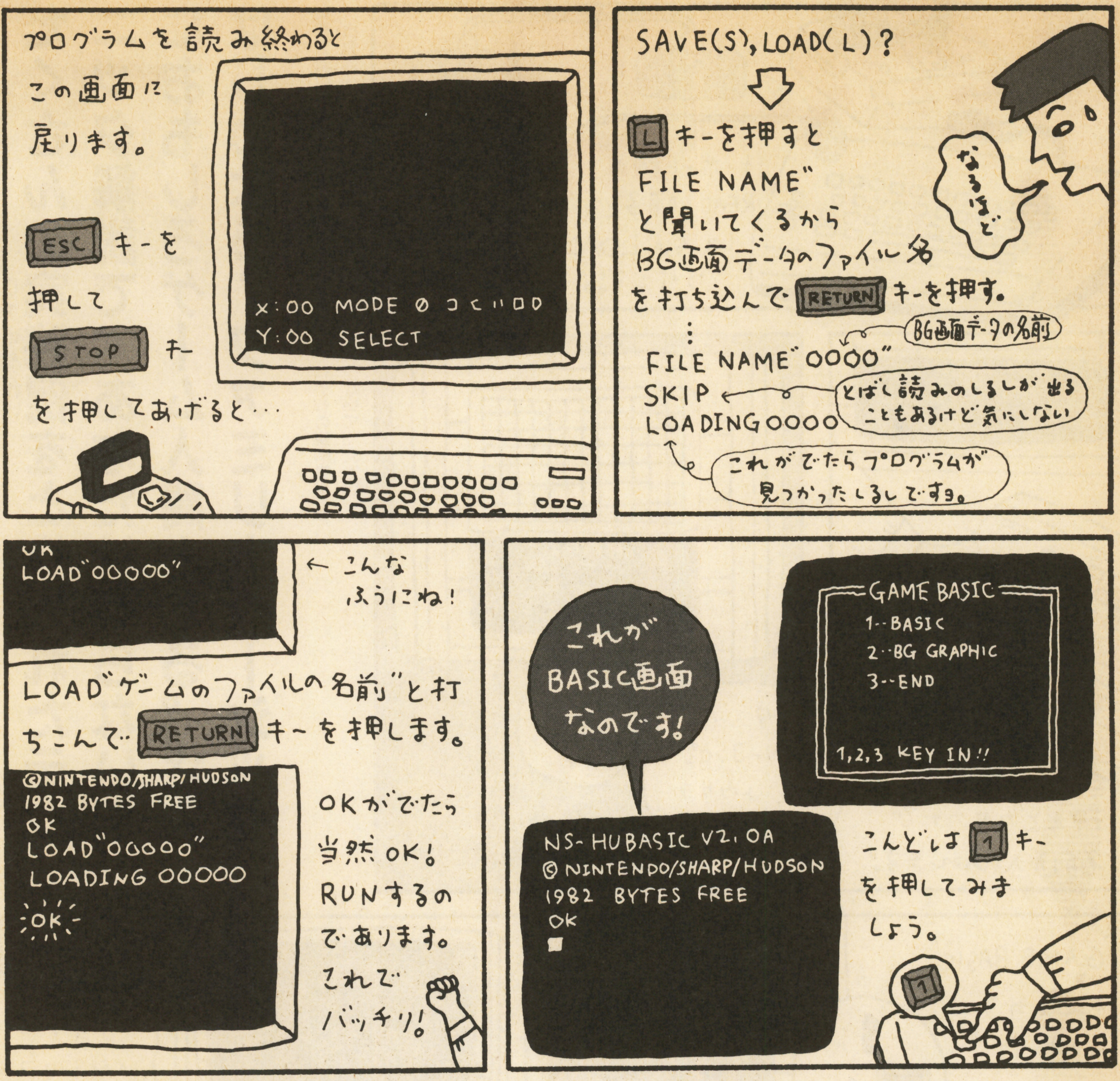
※任天堂以外のカセットレコーダのときは、それぞれの説明書を見て、出力端子(EARと書いてあることが多い)につないでください。







(次ページへつづく)



おもしろゲーム大作戦
テープリスト

ファミリーベーシック特訓ゲーム
（BG名：TYPE␣BG・プログラム名：TYPE）
ガレバレお姉さん!! 炎のビープ番
（BEEPBANBG・BEEPBAN）
気分はもうデバッカーバグシューター
（BUG␣BG・BAGSHOOT）
なみだのジューススタンド
（JUICE␣BG・JUICE）
マリオとルイージのVSバスケット
（BASKET␣BG・BASKET）
21（ファミンゴ風ブラックジャック）
（BJBG・BJ）
ルイージの3段とび
（3ダントビBG・3ダントビ）
一発勝負！ハイ&ロー
（HIGHBG・HIGH）
マリオの走り幅とび
（BG␣HABATOBI・HABATOBI）
カジノもまっ青？スロットマシン
（SLOTBG・SLOT）
レディの走り高とび
（HIGHJUMPBG・HIGHJUMP）
シューテングゲーム
（ ── ・SHOOTING）
数字をあてろ！
（MASTER−MINDBA・MASTER−MIND）
ファミリーおもちゃ箱
（オモチャBG・オモチャ）
ペンジンのペンギンズバー
（ ── ・PENGUIN）
色出し活用パレットツール
（ ── ・PALETTOOL）
かっとびモトクロッサー
（MOTO␣BG・MOTO）
VS 2次元ボクシング
（BOXING␣BG・BOXING）
むかしなつかし VSテレビピンポン
（PING␣BG・PINGPONG）
高速シューティング・スペースゲーム
（SPACE␣BG・SPACE1, SPACE2, SPACE3）

●本書の内容についてのお問い合せは、往復ハガキか封書（返信用封筒同封）で左記までお願いします。
〒102東京都千代田区四番町2－1
㈱日本ソフトバンク出版部
「ファミリーベーシックおもしろゲーム大作戦」係

初級編

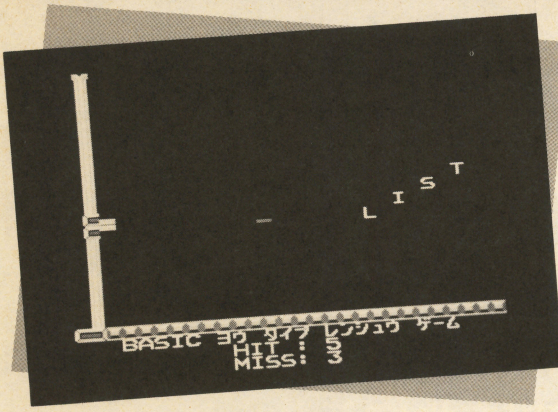
さあ、いよいよゲーム大作戦の開
始。まずは初級編で基礎を学ぼう

初級編

第1章

レッツ! ファミリーベーシック

まずはプログラム入力のABCを学んでみよう



今や一家に一台は必ずあると言われ、これがないと話についていけなくなり、村八分にされるファミリーコンピュータ。ゲームカートリッジで遊ぶだけでなく、リッパなベーシックでビシバシプログラムすることもできるというスグレモノだ。自分のオリジナルゲームを形にできるファミリーベーシック。うまく使いこなせるかな?

(1)ファミリーベーシックを動かそう

まずは必要なものの説明から。ファミリーコンピュータ、キーボード、ベーシックカートリッジ、この3つがどうしても必要なのは当然のこと。その他には、プログラムをセーブ・ロードするカセットテープレコーダが必要になる。あとは、根気とヤル気があれば、ファミリーベーシックを使いこなすことができるはずだ。

まずは、ファミリーコンピュータをTVと接続。次に電源を接続する。このとき、他のゲームカートリッジなどを動かしてみて、ファミリーコンピュータの動作を確認しておくとよいだろう。

ファミリーベーシックのカートリッジには、電源を切っている間もプログラムを保存しておけるようにバックアップ機能がついている。裏に単3電池2本を入れておけば、プログラムの途中でも電源を切ってひと休みできるという寸法だ。プログラムのバックアップはカートリッジの前のバックアップスイッチで行えるが、プログラムが入っていないときはOFFにしておこう。

いよいよカートリッジを差しこむ。そして、キーボードをつなぐ。さあ、電源ONだ。すべてOKなら、7色に輝くファミリーコンピュータのメインパネルがTV画面上に現れるはず。そこでキーボードのスペースバーを押すと……、あとは、ファミリーコンピュータの質問に答えて、ベーシックを選び出せばいいだけだ。

(2)まずは打ちこみ方から…

さてと、無事にファミリーベーシックを走らせることに成功したキミ。まずはオメデトウを言わせてもらおう。これでキミはこの後ドロ沼のように続くプログラミング地獄に足を踏み入れてしまったわけだ。もう足抜きはできないぞ。

ベーシックは動いているけれど何やら画面にOKと出たまま、四角がチカチカしているだけ。これじゃ何もおもしろくない。そう、ファミリーベーシックはプログラムを入力しないかぎり、ナーンにもしてくれないのだ。これがゲームカートリッジとは違うところだ。カートリッジポンでハイ、ゲームとはいかないというわけだ。

では、何のためにいちいちプログラムを入れてやるというメンドウなことをするのだろう。それは、既成品でない自分だけのオリジナルゲームを組めるようにするためなのだ。キャラクターの動き、ゲームのルール、背景などを自分で考え、それをTV画面上に実現していく。それがファミリーコンピュータでは可能なのだ。

プログラムの入力法だが、これはキーボードを押して1文字ずつ入力していくのが一般的。雑誌などにのっているファミリーベーシックのプログラムリストを見ながら、キーボードより入力する。最初は1文字ずつ確認して入力し、行の終わりには、その行全体をもう一度見返してRETURNを押す。確実に入力するのが重要だ。

(3)他人のプログラムでお勉強

ファミリーベーシックでプログラムをしようとする人は、まずその命令と文法を覚えなくてはならない。英会話の勉強だって最初は基礎となる単語や熟語、文法を覚えることが重要だ。まず、全体を組み立てている部品ひとつひとつの意味を知る必要がある。

それには、まずマニュアルを読むことだ。ファミリーベーシックのマニュアルには、各命令の説明が例題プログラムといっしょにつけられている。簡単な説明だが、命令の特徴をスッキリと示していて、初めてベーシックに触れる人でも混乱せずに理解できると思う。あせらずに、1命令ずつ自分のものにしていくのが一番だ。



命令のだいたいがつかめたら、今度は雑誌やマニュアルに付いているプログラムを打ちこんで、動かしてみよう。どんな命令が、どこで、どう使われているか。他人のプログラムを読めるようになれば、ベーシックを使いこなせるようになる日も間近だ。プログラムをキーボードで打ちこむことでタイプにも慣れておこう。

ファミリーベーシックタイプ特訓ゲーム

```
10 CGEN3:SPRITEON:VIEW:DIMM(1,4):PLAY"T1R0"
20 LOCATE3,21:PRINT"BASIC ﾖｳ ﾀｲﾌﾟ ﾚﾝｼｭｳ ｹﾞｰﾑ"
30 LOCATE 10,22:PRINT"HIT : 0";
40 LOCATE10,23:PRINT"MISS: 0";
50 Z=0:Y=1:X=2:I=-Y
60 DEFSPRITEZ,(Z,Y,Z,Z,Z)=CHR$(Y)+CHR$(188)+CHR$(Y)+CHR$(189)
70 DEFSPRITE7,(Z,Z,Z,Z,Z)=CHR$(237)
80 PLAY"O5C3DE":J=11:I=I+Y
90 J=J-Y:IFJ=Z THEN80
100 RESTORE110:FORA=0TORND(24):READA$:NEXT:G=Z
110 DATA BEEP,CGEN,CLEAR,CLS,COLOR,DIM,END,FOR,GOTO,IF,INPUT,LIST,LOAD,MOVE,NEW,
NEXT,PALET,PLAY,PRINT,REM,RUN,SAVE,STOP,THEN
120 FORA=1TOLEN(A$):DEFSPRITEA,(Z,Z,Z,Z,Z)=MID$(A$,A,Y):NEXT:A=A-Y
130 FORB=Z TOA-Y:M(Z,B)=232:M(Y,B)=103:SPRITEB+Y,M(Z,B),M(Y,B):NEXT:V=Z:W=Z
999 ON RND(5)+Y RESTORE 2000,2010,2020,2030,2040
1000 READS,T,U:T=T*((I AND3)+1)
1010 V=V+T:W=W+U:S=S-Y
1020 C=M(Z,Z)+V:D=M(Y,Z)+W
1030 FORB=Z TOA-Y:E=M(Z,B):F=M(Y,B):M(Z,B)=C:M(Y,B)=D:IFB>=G THENSPRITEB+Y,C,D
1040 C=E:D=F:NEXT:SPRITEZ,24,M(Y,G):IFM(Z,G)<32THEN3000
1050 H=ASC(INKEY$):IFH=Z THEN1100
1060 IFH=ASC(MID$(A$,G+1,1))THEN1500
1070 FORB=48TO240STEP24:SPRITE7,B,M(Y,G)+4:NEXT:SPRITE7
1080 PLAY"O1EC":M=M+1:LOCATE 15,23:PRINTM;
1100 IFS THEN1010
1200 GOTO1000
1500 FORB=32TOM(Z,G)STEP24:SPRITE7,B,M(Y,G)+4:NEXT:SPRITE7:DEFSPRITEG+Y,(Z,Z,Z,Z
,Z)=" "
1510 PLAY"O4CD":N=N+Y:LOCATE15,22:PRINTN;:G=G+Y:IFG<A THEN1100
1520 GOTO90
2000 DATA1,-8,0,2,0,8,4,0,-8,4,0,8,4,0,-8,4,0,8,4,0,-8,999,0,0
2010 DATA1,-8,0,999,0,0
2020 DATA1,-8,-8,3,0,0,1,0,16,3,0,0,2,0,-8,1,0,32,1,0,0,1,0,-24,2,0,0,1,0,8,2,0,
-8,999,0,0
2030 DATA1,-8,-4,3,0,0,1,8,8,2,0,0,1,-16,0,1,8,0,5,0,0,1,-8,-16,999,0,0
2040 DATA1,-8,0,2,0,0,2,8,4,2,-8,0,2,0,-4,2,-4,0,3,0,-4,4,0,4,999,0,0
3000 PLAY"O2C4CD"
3010 LOCATE11,10:PRINT"GAME OVER":END
```



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0		E10																										
1		E20																										
2		E20																										
3		E20																										
4		E20																										
5		E20																										
6		E20																										
7		E20																										
8		E20																										
9		E20																										
10		E20																										
11		E20																										
12		E20																										
13		E20																										
14		E20																										
15		E20																										
16		E20																										
17		E20																										
18		E20																										
19		E20																										
20	A10	A00	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10	F10

⑷ゲーム入力法

この本の目的は、あくまでファミリーベーシックを使いこなせるように、勉強する手助けをすることだ。プログラムは、できるだけキーボードから手で打ちこもう。自分の思った命令をすぐ打ちこんで、動きをその場で確かめるのが理解を早めるのにたいへん有効だからだ。

新しいプログラムを入力するときには、まず、

NEW↵

を行うことから始めよう。こうすると前に入っていたプログラムがきれいさっぱり消えて、新しいプログラムを入力する場所ができる。

次にプログラムを入力するわけだが、プログラムリストは、まず行番号と呼ばれる数字で始まり、それに続く命令からなっている。それがプログラム1行である。1行入力するごとにRETURNを押し、リストどおりに入力しよう。

次に、BGグラフィックスを使って背景を入力する。BGグラフィックスについての説明は、第2章にゆずることにするが、マニュアルで入力法を見てほしい。

⑸ゲームの遊び方

ベーシックの命令文のスペルをすばやく打ちこめるようにするため、ベーシック入門用タイプ練習プログラムを作ってみた。ただの練習プログラムではおもしろくないので、ゲーム仕立てとした。

ルールは簡単。画面の右端からジワジワと迫ってくるアルファベットを次から次へと、キーボードを押しまくって撃滅するというものだ。正しいアルファベットのキーを押すと画面左端の砲塔が動いて、アルファベットを撃ち落としてくれる。撃ち落とせるアルファベットは、先頭のものだけ。アルファベットは、ベーシックの命令のスペルの順番にやってくる。命令10個分のアルファベットを撃ち落とすと、1ラウンド終了。ラウンドが進むごとに動きが速くなったり、変化したりするようになっている。キーを打つのが間に合わなくて、アルファベットが左端まで来たらゲームオーバーだ。砲塔が動いていたり、弾丸が飛んでいる間は、次の入力はできないぞ。キミはどこまでいけるかな?

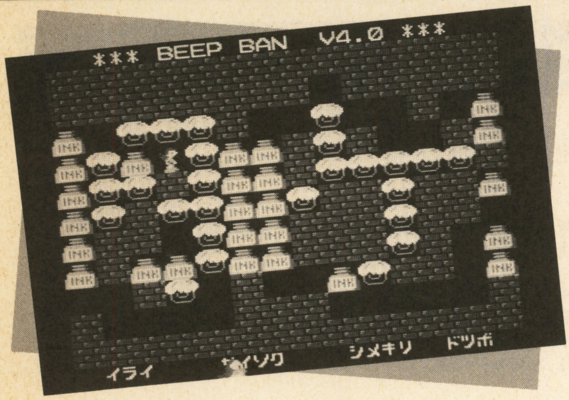
ゲームスタートはRUN↵だ。



初級編

第2章 BGグラフィックを使おう!

BGグラフィックを使いこなそう



ファミリーベーシックの特徴のひとつ、背景を描くBGグラフィックス。少ないメモリでプログラムするには、なくてはならないツールだ。背景を作るだけではなく、チョットしたお絵描きツールとして、ゲームの案を練るときなどにも使えるぞ。

BGグラフィックスの使い方をバッチリマスターしてしまおう。

(1)BGグラフィックスって何?

バックグラウンドグラフィックス。つまり背景を描くためのツール。これが名前の由来である。けっこう使いがってもよい。まずは、マニュアルを読み返して、基本的な使い方をおさらいしてほしい。

BGグラフィックスは、ベーシックより、

SYSTEM⏎

とした後に、メニューで選択できるようになっている。

四角枠のカーソルが、カーソル移動キーで上下左右に動く、これがペンの代わりだ。その位置に、キャラクターテーブルB(マニュアル参照)のキャラクターを置いて、絵を作っていけるわけだ。

BGグラフィックスには、いくつかのモードがあって、ESC(エスケープキー)で選択できる。

●セレクトモード…キャラクターを選択してカーソルの位置にセットする。決定にはスペースバーを使う。またRETURNキーで、色を変えることもできる。

●ムーブモード…すでに画面上に作った絵を移動させるモード。INSで取りこんでカーソル移動し、DELでセットする。

●コピーモード…同じキャラクターを次々と描きこんでいくとき便利なモード。INSキーでとりこみ、DELキーでセットできる。

●クリアモード…全画面の絵を消して描き直すとき用いる。

●ファイルモード…BG画面を、カセットにセーブしたりロードしたりする。

●キャラクターモード…キーボードより、文字をカーソル位置に入力する。

(2)ベーシックでのBG画面の使い方

BGグラフィックスからベーシックにもどるには、ESCキーを押して、モード選択モードになったとき、STOPキーを押し、ベーシックを選択すればよい。

BGグラフィックスで描いた背景は、ベーシックにもどると画面上から消えてしまう。しかし、心配無用。

VIEW⏎

と入力すれば、たちまち画面に現れてくる。ゲームスタートする前にこの命令を入れておけば、BGグラフィックスで描いた背景でゲームができるというわけだ。

ファミリーベーシックの画面は、4枚の画面が重なってできている。

一番奥は背景色の画面。これはPALET文で指定でき、画面全体を好きな色に変えられる。

次が背景の後ろのスプライト用画面。この画面に設定されたスプライトは、背景と重なったときは、その影に隠れて見えなくなる。遠近感を出すために使われたりする。

そして次が、背景用の画面。VIEW文を実行することで、BGグラフィックスで描いた背景がこの画面に転送される。PRINT文による出力も、この画面に対して行われている。

一番手前は、背景の前のスプライト用画面だ。たいていのスプライトは、この画面上で動く。背景と重なっても消えることはない。背景の前後の指定は、DEFSPRITE文で行うことができる。

(3)BGグラフィックスの利点・欠点

さて、BGグラフィックスの長所と短所を見てみよう。

まず、メモリの問題。ファミリーベーシックはV2・0で2K弱、V3でも4K位のメモリしか、プログラムに使えない。こんななかに画面の背景を描くPRINT文などを入れたら、それだけでメモリが足りなくなってしまう。BGグラフィックスのメモリは、プログラムするメモリとは別のところにあるので、少ないメモリをすべてプログラムに利用することができる。省メモリには、とても有効なツールだ。しかし、メモリエリ

アが違うため、ベーシックのバックアップ機能が効かない。電源を切ったら消えてしまうのである。カセットにセーブしておく必要があるわけだ。

それともう1つの欠点は、下3行には背景を描けないということだ。ベーシックの処理で得点などを表示するエリアとして利用するのが無難な解決法である。

⑷ゲーム入力法

このプログラムは、メモリをギリギリまで使っているので、余分なスペースをできるだけつめてある。リストのとおりに打ちこんで、まずはセーブしておこう。

そして次にBGグラフィックスでパズルの面を入力する。これもセーブしておいてほしい。

⑸BGツールによる面の入力法

このゲームでは、BGグラフィックスをパズルのコンストラクションセットのように利用してみた。つまり、BGグラフィックスの内容を見てプログラムがパズルを作るということだ。BG画面ひとつにつき、4つのパズルが入力できる。入力にはBGグラフィックスのキャラクターモードを使い、数字の1が壁、2がライター、3がインク、4がお姉さんと、パズル

ガンバレお姉さん！炎のビープ番

```
100 CLS:PRINT"SELECT";:Q=&H0:R=&H1:C=&H2:T=&H8:A=STICK(Q):IFA=Q THEN100
110 READJ,M,U,V,W,H,I,G:DATA0,0,26,0,-2,19,2,-2,14,10,0,26,2,3,19,2,14,0,0,26,2,
19,2,-2,0,10,26,0,-2,3,19,2
120 A=A/C:B=B+R:IFA THEN110
130 PRINTB:INPUT"T";S:PALETS C,32,32,16,23:VIEW:SPRITEON:FORA=U TOV STEPW:RESTOR
E170:B=H-G:GOSUB470:FORB=H TOI STEPG:Z=VAL(SCR$(A/C+J,B/C+M)):IFZ=4THENX=A:Y=B:Z
=Z AND&H3
140 ONZ+R RESTORE160,170,180,190:IFZ=&H3THENK=K+R:DATAT3M1V7Y0:T3M1V7Y0:T3M0V15,
O4C3RCR:O3E3RER:O2C7,O3BRAR:DRER:G,G4A2B4O4V12C8:F4E2D4V12C8:DC2,@
150 GOSUB470:NEXT:RESTORE170:GOSUB470:NEXT:G=Q:I=Q:IFX+Y THEN170:DATAT3M1V15Y0:T
3M1V15Y0,O2C3D#DD:O1C7,C5O1G:O0G5G,#G3GBO2C:#CD,Y3C9:Y3#D,@
160 PRINT"LADYノ イチ ハ?":END:DATA32,32,32,32,1
170 LOCATEC,23:PRINT"イライ      サイソク      シメキリ   ト゛ツホ゛";:H=&H3:DEFMOVE(Q)=SPRITE(&H4,H,
S,112,Q):W=X*T+16:Z=Y*T+24:GOSUB410:POSITIONQ,Q,208:MOVEQ:DATA195,195,195,195,1
180 IFK=I THEN490:DATA216,218,224,225,1
190 Z=STRIG(Q):IFMOVE(Q)=Q ORZ=R THEN540:DATA249,251,250,252,0
200 IFZ<>&H4THEN190:DATA243,247,245,248,0
210 Z=STICK(Q):IFZ=Q THEN190:DATA52,1
220 ONZ RESTORE290,300,310,320,330,340,310,350,360,370:READH,U,V:A=X:B=Y:GOSUB35
0:J=ASC(SCR$(A,B)):IFJ=32THEN400:DATA28,2
230 IFJ=195THEN550:DATA48,1
240 GOSUB350:F=ASC(SCR$(A,B)):IFJ<>249THEN280:DATA28,1
250 RESTORE190:IFF=32THEN320:DATA52,2
260 IFF=216THENRESTORE200:I=I+R:GOTO320:DATA32,2
270 GOTO550:DATA48,2
280 IF J<>243 THEN550:DATA32,1
290 RESTORE200:IFF=32THENRESTORE190:I=I-R:GOTO320:DATA3,2,0
300 IFF=216THEN320:DATA7,-2,0
310 GOTO550:DATA0,0,0
320 GOSUB470:GOSUB390:IFJ=243THENRESTORE180:GOTO340:DATA1,0,2
330 RESTORE160:DATA2,2,2
340 GOSUB470:GOTO400:DATA8,-2,2
350 A=A+U:B=B+V:DATA5,0,-2
360 IFA<Q THENA=26:DATA4,2,-2
370 IFA>26THENA=Q:DATA6,-2,-2
380 RETURN
390 A=A-U:B=B-V:GOTO360
400 W=X*T+16+U:Z=Y*T+24+V:FORA=R TOT:GOSUB410:W=W+U:Z=Z+V:NEXT:A=X:B=Y:GOSUB350:
X=A:Y=B:GOTO180
410 G=R-G:ONH+G*T RESTORE210,220,220,220,230,240,240,240,250,260,260,260,270,280
,280,280:READF,E:ONE GOTO420,430
420 DEFSPRITEQ,(C,R,Q,Q,Q)=CHR$(F)+CHR$(F+R)+CHR$(F+C)+CHR$(F+&H3):GOTO440
430 DEFSPRITEQ,(C,R,Q,R,Q)=CHR$(F+R)+CHR$(F)+CHR$(F+&H3)+CHR$(F+C)
440 IFW<T THENW=232:GOTO460
450 IFW>232THENW=T
460 SPRITEQ,W,Z:RETURN
470 LOCATEA,B:READD,E:PRINTCHR$(D);CHR$(E);:LOCATEA,B+R:READD,E,L:PRINTCHR$(D);C
HR$(E);:COLORA,B,L:RETURN
480 DATA32,2
490 CLEAR:LOCATE8,10:PRINT"O K !":RESTORE140:ERAQ
500 READA$:IFA$="@"THEN520
510 PLAYA$:GOTO500
520 IFSTRIG(Q)<>&H1THEN520
530 RUN
540 CLEAR:LOCATE8,10:PRINT"LATE.":RESTORE150:GOTO500
550 PLAY"O1COC":GOTO180
```

に登場するキャラクターを、すきなところにおくことができる。このBGグラフィックスの使い方は、背景をプリント文で変えられることと、VIEW命令を行うたびに最初の背景を呼び出すことができるというふたつの特徴を利用している。

⑥ゲームの遊び方

このゲームは、BEEP編集部の紅一点（こういってん）、ⓕのお姉さんの日常生活をプログラムしたものである。

目的は、インクを押していって、オーバーワークのライターに原稿（げんこう）を書かせること。インクとライターを重ねると、原稿になり、全部のインクを原稿に変えることができればOKだ。しかし、この業務は時間との勝負、画面下には、こくこくとシメキリまでの時を刻み続ける「仏のペンジン」が…。

お姉さんは、上下左右とナナメに移動できる。✚ボタンで移動方向を決めてBボタンを押すとその方向に1歩進むようになっている。ナナメの動きは、その方向の上下と左右のボタンを同時に押して、Bボタンを押すことになる。

お姉さんは、インクを押すことはできるが引くことはできない。そして、原則としてライターの上を通ることはできない。ただし、インクを押していって、ライターに重ね、原稿にした後、その原稿を動かしてふたたびライターとインクにしたときのみ、ライターを踏（ふ）んで通過することができる。（原稿はボツで、ライターはオシオキされるということ。ア～、コワ！）

ゲームがスタートするとパズルの面を選ぶことになる。✚ボタンで右1、左2、下3、上4の面を選び、次に1～255の数字でペンジンの進む速さを入力する。

お姉さんはインクビン1個しか押すことはできない。また、画面の左右はつながっている。

BGグラフィック画面データ

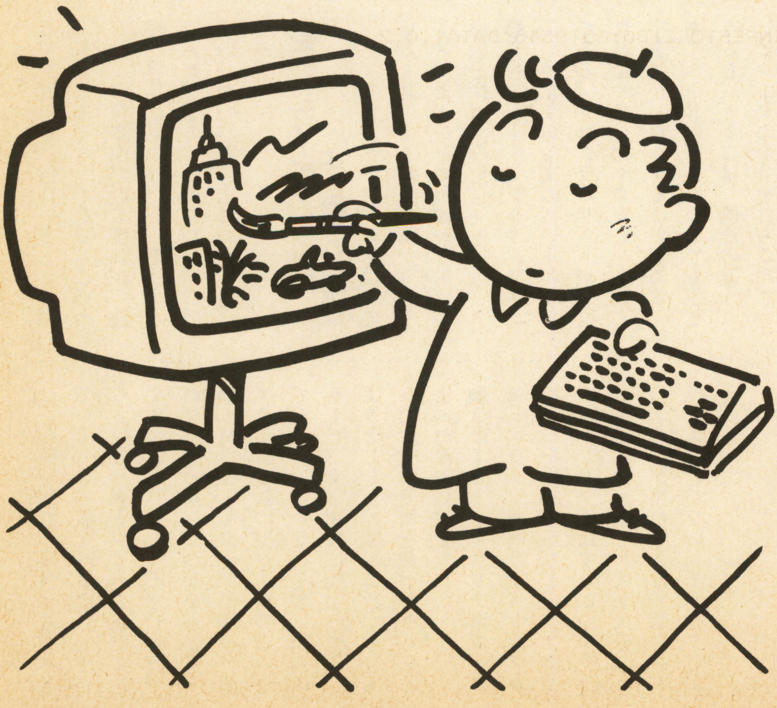
1面目のパズル（1～9行・0～13列）／3面目のパズル（1～9行・14～27列）／4面目のパズル（11～19行・0～13列）／2面目のパズル（11～19行・14～27列）

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0				*	*	*		B	E	E	P		B	A	N			V	4	・	0		*	*	*			
1					1	1	1	1	1	1		1	1		1					1	1	1	1	1	1		1	1
2	1	1			3		3			3				1	1		1	1	1				2	3			2	1
3	1	2	1		3										1		1		1		1	1	2	2				1
4	1	1			1	1		1	1		1	1			1	2	1		3	3	3	1	1	2	1	3	3	1
5	1	2	1	3	1	2		1	2		1	2	1		1		1		3	4	3	2	2	2	1		3	2
6	1	2	1		1	1		1	1		1	1			1		1		3	3	3	1	1	1	1			1
7	1	1			1	2		1	2		1	4	3		1		1	3	1		3		2		2		1	1
8					1	1	3	1	1		1	3			1			1	2	1	1	1	1	1	1		1	1
9	1	1	1	1									1	1	1		2	2				3				3	2	1
10																												
11	1	1	1	1	1	1	1	1	1		1			1		1	1	1	1		1	1		1	1	1		
12				2	2	2	2	2				1		1	1			3		4	1	1			1	3	3	1
13	1	2		1	1	1	1	2	1		1	1		1	1			1	1	2	1	1		1		1		1
14	1	2	1		1		1	1						1	1	1	1	1			1	2		3	1	3	3	3
15	1	3	1		1			1			1	1	1	1		2	1					1		1	1	1	3	1
16	1		1			1			1	1	1	1	1			1	1		1	2	1	1	3	1			1	1
17	1		1	1	1	1	1	1				3	3		1	1				1				1		1	1	
18	1					3			3	3	3	3		1				2	1			1	1	1	2	1		1
19	1	1	1		1	1	1	1	1	1				4			1	1				1	2	2			1	2
20																												

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0																												
1																												
2																												
3																												
4																												
5																												
6																												
7																												
8																												
9																												
10																												
11																												
12																												
13																												
14																												
15																												
16																												
17																												
18																												
19																												
20																												

▶これがパズル作成用紙だ。太いワクの中が1面分。1～4の数字をキャラクターモードで入力してパズルを作ることができる。

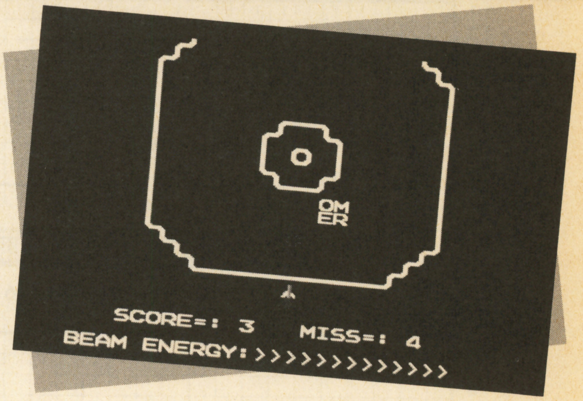


初級編

第3章
エラーを撃退(げきたい)!

プログラマー最大の敵、エラーの解消法

プログラマーの敵は、貧乏、寝不足、胃潰瘍と山ほどあるが、そのなかでも一番恐ろしいのが、エラー(またの名を虫)である。やっとのことで作ったプログラムを、打ちこんでRUN↵…、ところがまったく動かない。または、自分の計画したのとは違った動作をしてしまうなどなど。プログラムはまさに虫との戦いなのである。



(1)バグとは何ぞや?

では、プログラマーの永遠の宿敵、「虫(バグ)」の生い立ちをプログラムを作る過程にそって見ていこう。

まずプログラマーは、自分の組むプログラムに何をさせたいかを決め、その方針を定める。つまり、プログラムの機能を決めるのである。この段階で生じるバグは、コンピュータの性能と見合わない要求をプログラマーがすることで生じる。たとえば、とてもベーシックの処理速度ではこなせない高速処理を要求したり(ベーシックでギャラガーを作ろうとしたり)、とてもメモリに入りきらないようなデータ処理を要求したり(何十画面もあるグラフィックアドベンチャーを作ろうとしたり)することから生じる。この種のバグは、見落とすと大変である。プログラムを作っていくうちにどんどん大きくなり、最初からやり直しという悲劇(ひげき)的結果になる。

次にプログラマーがよくやるミスは、公式や計算式を間違えて使ってしまうことによって生じるバグ。こいつもなかなかやっかいで、そう簡単にはシッポを出さない。これら2つは考え方のバグである。

これに対して、もっともよく生じやすいのが、打ちこみ時に生じるバグである。俗にいうところの「打ちこみミス」である。こいつらは、多種多様で数も多いが、比較的容易に見つけ出し、退治することができる。この章では、この手のバグの処理方法について述べてみよう。

(2)ファミリーベーシックによくみられる虫は

ファミリーベーシックを扱っているとき、よくお目にかかるエラーメッセージを4つばかり挙(あ)げて、原因を探(さぐ)ってみた。

●SNエラー…シンタックスエラー、つまり命令文のスペルのミスである。プログラムリストが2段にわたっていたりすると、よく文字を抜かしたりすることがある。キーボードのミスタッチでも生じうるし、ときには、機種の違うコンピュータの命令を打ちこんでいたりすることもある。

●ILエラー…イリーガルファンクションコールエラー。命令に数値や変数で位置や長さなどを与えてやるとき、その命令が実行不可能な値であることを示すメッセージである。よく生じるのは、数値変数で、命令に値を与えていて、その値を計算違いしているというパターン。その場でその変数の値をPRINT文で確認すれば解決の糸口がつかめるかも。



●TMエラー…タイプミスマッチエラー。変数の型が1つの式の中でかみ合っていないときに出るエラーで、文字変数に数字を代入しようとしたり、関数に数字を与えるのに文字変数を使ったりしてしまったときに生じる。

●OMエラー…アウト・オブ・メモリエラー。チョット大きなプログラムを作っていると、ファミリーベーシックではすぐこのエラーにお目にかかる。リスト入力でこのエラーが出たときは、余分なスペースなどがないかチェックしよう。

気分はもうデバッガー・バグシューター

```
100 PLAY"T1"
110 VIEW:CGEN&H3:SPRITEON:CGSET&H1,&H1
120 LOCATE&H0,21:PRINT"   SCORE=: 0   MISS=: 0"
130 PRINT: PRINT"BEAM ENERGY:>>>>>>>>>>>>>>>";
140 PALETB&H0,31,48,48,48
141 PALETB&H1,31,31,31,48
150 PALETB&H2,31,31,31,31
160 PALETB&H3,31,31,31,31:S=&H1
170 PALETS&H0,31,16,&H0,&H1
180 PALETS&H1,31,22,22,22
190 PALETS&H2,31,48,22,&H7
200 PALETS&H3,31,48,22,&H7
210 DEFSPRITE&H0,(&H0,&H0,&H0,&H0,&H0)=CHR$(212)
220 DEFSPRITE&H1,(&H1,&H0,&H0,&H0,&H0)=CHR$(211)
230 DEFSPRITE&H2,(&H1,&H0,&H0,&H0,&H0)=CHR$(221)
240 DEFSPRITE&H3,(&H1,&H0,&H0,&H0,&H0)=CHR$(221)
250 DEFSPRITE&H4,(&H2,&H1,&H0,&H0,&H0)="SNER"
260 DEFSPRITE&H5,(&H3,&H1,&H0,&H0,&H0)="OMER"
270 A=124:B=&H0:C=B:D=B:E=B:P=B:U=B:V=15
280 I=B:J=B:K=B:M=B:N=B:O=B
290 P=&H1-P:Q=&H1-Q:SWAPB,D:SWAPC,E:SWAPI,M:SWAPJ,N:SWAPK,O
300 GOSUB540:F=STICK(&H0):IF(F AND &H1)ANDA<240THEN A=A+&H8
310 IF(F AND &H2)ANDA>16THENA=A-&H8
320 SPRITE&H0,A,167:SPRITE&H1,A,175
330 IFB THEN430
340 T=T/&H2:IFV<&H4 ORT<>&H0 ORSTRIG(&H0)=&H0THEN370
350 LOCATE&H8+V,23:PRINT"    ";
360 T=&H2:V=V-&H4:B=A:C=159:SPRITEP+&H2,B,C:GOTO430
370 IFI THEN I=I+(K AND&H3)*&H4-&H6:J=J+(K/&H4)*&H4+&H4:SPRITEQ+&H4,I,J:GOTO400
380 IFRND(10)<&H4THENI=128:J=96:K=RND(&H8):GOTO400
390 GOTO420
400 IFABS(A-I)<12 ANDABS(167-J)<&H5THEN630
410 IFI<16ORI>240ORJ>180THEN470
420 GOSUB660:GOTO290
430 IFABS(B-I)<12 AND ABS(C-J)<&H7 THEN580
440 IFABS(B-M)<12 AND ABS(C-N)<&H7 THEN570
450 B=B-&H2*SGN(B-124):C=C-&H4:IFC<80THENB=&H0:SPRITEP+&H2:GOTO370
460 SPRITEP+&H2,B,C:GOTO370
470 I=&H0:R=R+&H1:SPRITEQ+&H4:LOCATE21,21:PRINTR;:PLAY"O2C1CD":IFR<10THEN420
480 LOCATE&H5,&H1:PRINT"*****GAME OVER*****"
490 FOR A=&H0TO&H10:PLAY"O0C0D":PALETB&H0,31,38+A,31,48:NEXT
500 LOCATE&H6,&H2:PRINT"PUSH START BUTTON"
510 FOR A=&H0TO&H10:PLAY"O0C0D":PALETB&H0,31,48-A,31,31:NEXT
520 IFSTRIG(&H0)<>&H1THEN520
530 RUN
540 W=(W+&H1)AND&H3:IF W=&H0 AND V<15THENLOCATE12+V,23:PRINT">";:V=V+&H1
550 IFP THEN PALETS&H1,31,38,38,38:RETURN
560 PALETS&H1,31,22,22,22:RETURN
570 Q=&H1-Q:SWAPI,M:SWAPJ,N:SWAPK,O
580 PALETS&H2+Q,31,48,48,48
590 PLAY"O4C2E"
600 I=&H0:SPRITE&H4+Q
610 PALETS&H2+Q,31,48,22,&H7:B=&H0:SPRITEP+&H2
620 U=U+&H1:LOCATE10,21:PRINTU;:GOTO370
630 FOR A=&H0TO10:PALETS&H0,31,22,31,48
640 PLAY"O0C2O4C":PALETS&H0,31,16,&H0,&H1:NEXT
650 GOTO480
660 PALETB S,31,31,31,31
670 S=S+&H1:IFS=&H4THENS=&H1
680 PALETB S,31,31,31,48
690 RETURN
```

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	K72					K71	L21			K73	K53	K53	K53	K53	K53	K53	K53	K53	L03			L11	L01					L02
1	K62				K71	L21			K73	L23									L13	L03			L11	L01				K62
2	K62				K61			K73	L23											L13	L01			K61				K62
3	K62				K61			K63			K72	K52	K52	K52	K52	K52	K52	L02			K63			K61				K62
4	K62				K61			K63		K72	L22							L12	L02		K63			K61				K62
5	K62				K61			K63		K62			K71	K51	K51	L01			K62		K63			K61				K62
6	K62				K61			K63		K62		K71	L21			L11	L01		K62		K63			K61				K62
7	K62				K61			K63		K62		K61		K70	L00		K61		K62		K63			K61				K62
8	K62				K61			K63		K62		K61		L10	L20		K61		K62		K63			K61				K62
9	K62				K61			K63		K62		L11	L01			K71	L21		K62		K63			K61				K62
10	K62				K61			K63		K62			L11	K51	K51	L21			K62		K63			K61				K62
11	K62				K61			K63		L12	L02							K72	L22		K63			K61				K62
12	K62				K61			K63			L12	K52	K52	K52	K52	K52	K52	L22			K63			K61				K62
13	K62				K61			L13	L03											K73	L23			K61				K62
14	K62				L11	L01			L13	L03									K73	L23			K71	L21				K62
15	K62					L11	L01			L13	K53	K53	K53	K53	K53	K53	K53	K53	L23			K71	L21					K62
16	L12	L02					L11	L01													K71	L21					K72	L22
17		L12	L02					L11	K51	K51	K51	K51	K51	K51	K51	K51	K51	K51	K51	K51	L21					K72	L22	
18			L12	L02																					K72	L22		
19				L12	L02																			K72	L22			
20					L12	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L22				

(3)除虫のやり方

この章では、リストの打ちこみのときに生じる虫の退治のし方について時期別に考えてみよう。

入力前…まず注意してもらいたいのが、これから打ちこもうとしている機種のリストが、本当に自分の持っている機種のリストかどうかということ。ファミリーベーシックにもV1、V2・0、V3と3種も出ている。ほとんどはV2・0用のプログラムだが、たまに、V3用プログラムが掲載されていることがあるのでご用心。

次に注意することは、ファミリーベーシック内に、前のプログラムが残っていないことを確認することだ。打ちこみ前にNEWを行っていればベーシックについては、大丈夫だが、CLEAR文など、特殊な命令を使ってベーシックのプログラムエリアが狭くなっていることも考えられる。新しいプログラムを入力するときは、バックアップスイッチをOFFにした状態でファミリーベーシックの電源をONにすれば、初期状態となり、確実だろう。

入力時…とにかく打ち間違いをしないよう、全力を尽くすべきである。リストに定規を当てたり、入力した行をラインマーカーでチェックするなどという方法もある。疲れてきたら、コーヒーブレークというのもミスを減らすのに有効だ。1字1字を確実に入力し、1行入力したら、また確認。時間がかかるようだが、結局それが一番早い方法なのである。

入力後…入力を終えたら、まずセーブ。そしてLOAD?で照合することだ。せっかく入力したプログラム、チョットしたミスでパーになってしまうのはおしい。間違っていても一応セーブしておこう。

実行時…RUN⏎の後、たいていの場合は、エラーメッセージが出る。その行内にリストと違っているところはないか、変数の値などが変なところはないかを調べて、他の行のなかで、それらの変数の出てくるところを再チェックするのがよい方法だ。

OMエラーや、動作がどことなく変なときは、もう一度、頭からリストとつけ合わせていくよりしかたがない。

除虫には、根気と運が必要だ。

(4)ゲーム入力法

プログラムリストを入力して、セーブする。次にBGグラフィックスを入力するのだが、このBGグラフィックスは、色の指定がされているので、セレクトモードのところのマニュアルをもう一度見ておいてほしい。BGグラフィックスのセーブもわすれずに。

(5)ゲームの遊び方

さてさて、バグとりがいかに大変なお仕事か、少しは理解していただけたと思う。せめて、ゲームのうえでは景気よく、バグ退治といってみよう。

キミは電子回路のなかを飛びまわるバグシューター。ウヨウヨ飛びまわるバグの大群を次々と退治してまわるのが任務。前方からやってくるバグを撃ち落とそう。

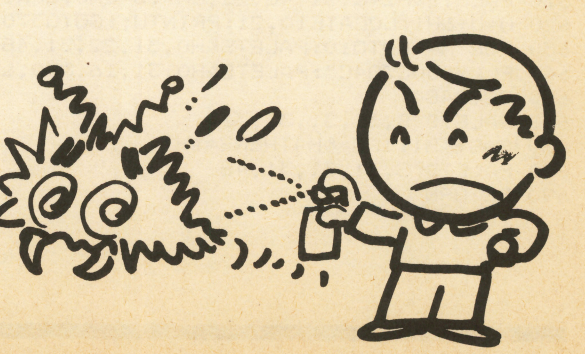
バグに衝突したり、後ろに10個以上のバグを逃がしてしまったときはゲームオーバーとなる。

RUN⏎

でゲームスタート。画面中央のやや下に出てきたのが、私らのバグシューター。✢ボタンの左右で移動し、Aボタンでバグ消滅弾を発射できる。画面下の∧印がエネルギーを示し、あんまりムダ弾ばかり撃ちまくると、エネルギー不足で弾が出なくなってしまう。

ゲームオーバーまで、キミは何個のバグを退治できるかな?

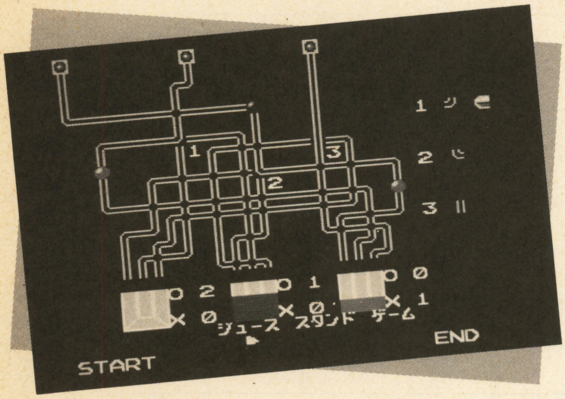
(実際のバグとりもこんな風にできると楽なんですが…。だれか、私のプログラムのバグを退治してくれませんか…)



初級編

第4章 ゲームを作ってみよう!!

いよいよプログラム!!



いよいよゲーム作りにチャレンジだ。ゲームのできていく過程を見ていきながら、アイディアの生かし方、作っていく手順などについて見ていこう。

そのほか、ゲーム作りに必要な特製のBGグラフィックス用のレイアウト用紙を作ってみたので、その使用方法の説明などもしてみたい。

(1)ゲーム作りのとっかかり

ゲームを作るきっかけとなることは意外と日常の生活の中にころがっていることが多い。

例えば雨のふる日曜日、窓からぼんやり外を見ている時、ひさしから落ちる水滴を見て、ゲームのアイディアが湧いてくることがあるといったふうである。

ジュースがパイプをつたって流れていく。そして、所々に切り替え機があってパイプをつなぎ替えることができるようにしておく。ジュースを決まったビンに集めることができればよし、だめならやりなおしというルールでゲームはできないだろうか……ファミリーベーシックなら、いろんな色が表現できるから、ジュースは3色ぐらいにして、別々のビンに集めることにしよう。切り替えは、3カ所ぐらいにして、順番にしかできないようにすれば、入力も簡単だし、パズル性も出てくる……。

といった具合に、日常のなにげないことがらから、ゲーム作りのネタは浮かび上がってくるものなのだ。このようなイメージは、誰でも作り出すことができるのだが、ゲームのアイディアとして固まる前に、どこかに流れていってしまうということがけっこう多いようである。イメージをすばやくつかまえて、ルールにあてはめてしまうことが、アイディアを生み出す重要な点であろう。

しかし、それをゲームにするには、もう1つきっかけがいる。例えば「シメキリ」のような……。

(2)まずは画面に表現してみよう

ゲームを作る気もあるし、プログラムテクニックも十分にある。しかし、今一歩ふみきれないという人もいると思う。そのタイプの人は、頭の中で問題をひねくりまわしすぎて、よけいに難しいものにしていることが多いようだ。まず頭に浮かんだイメージをすっきりと整理して、それに必要な最低限度の要素を取り出して画面に表現してみるのがよいと思う。

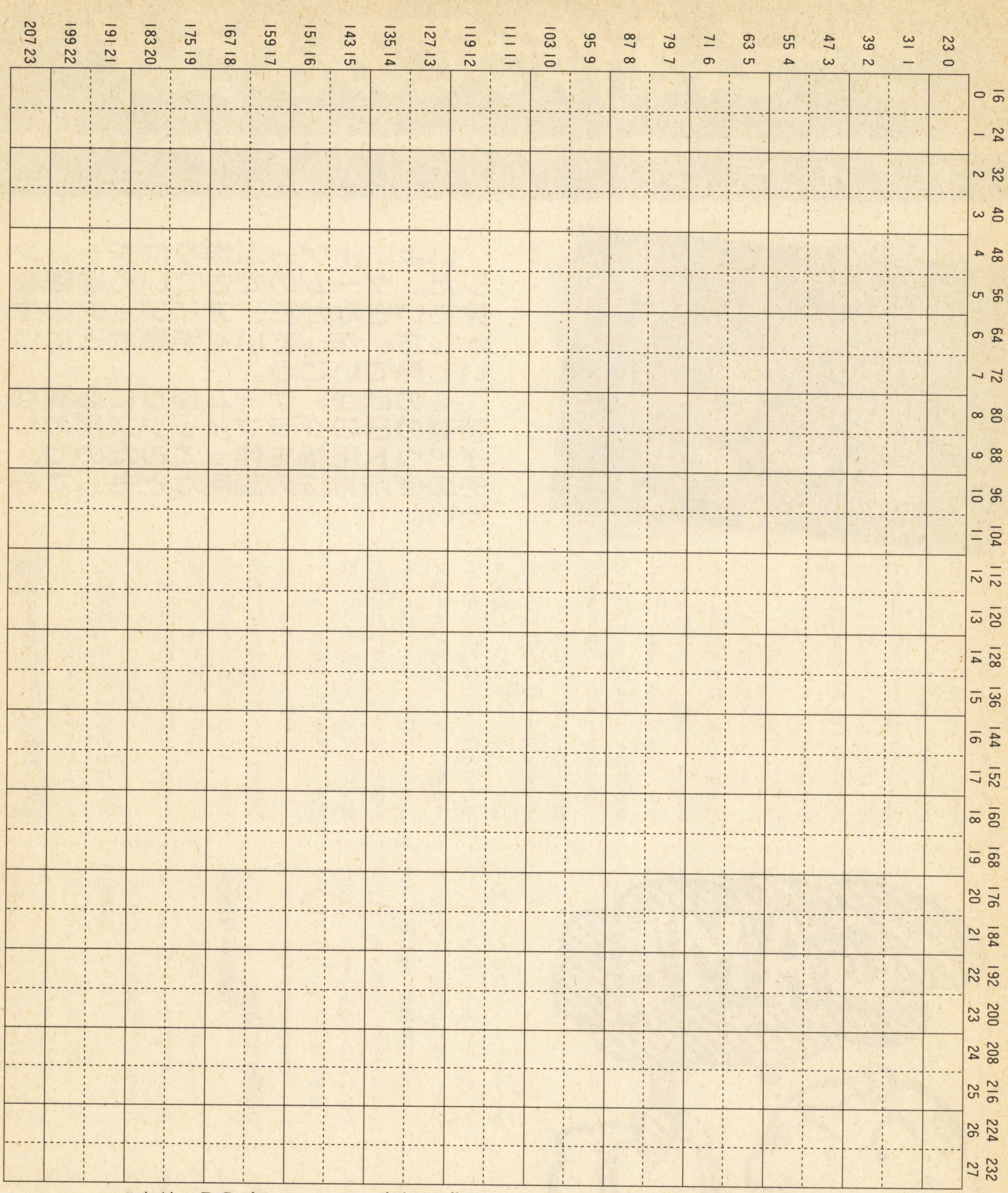
これは、まず第1にイメージを具体化するという意味でひとつの前進である。頭の中の考えは、すぐに変化していってしまうので、その時の状況を画面に表しておくのである。

第2に、それを見なおすことでさらに新しいイメージを自分の中に生じさせることができるということである。人間は自分の経験から、さらに新しい考えを生み出すことができる。

第3には、コンピュータの表現力の限界を身をもって知ることが



特製BG用紙



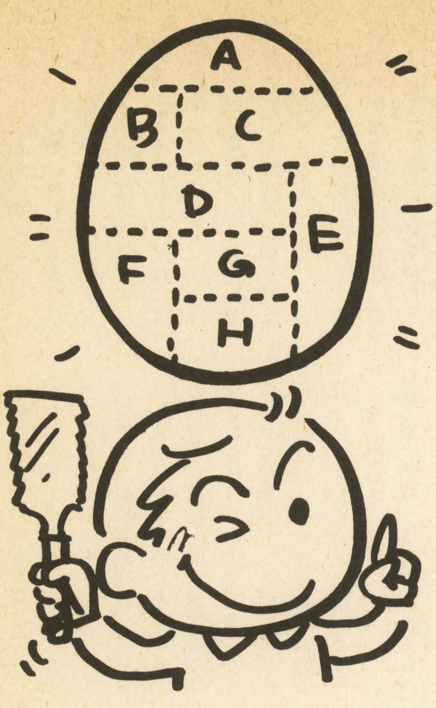
これは、BGグラフィックスで画面を作っていくのに便利なように工夫した特製BG用紙である。

1マスが1キャラクターが入る大きさで、縦横方向に、キャラクターでの位置（LOCATE文やBGグラフィックスで用いる座標）と、スプライトでの位置(SPRITE、POSITIONなどで用いる座標)とが、対応するように記入してある。

また、表の点線で区切られたキャラクターは、COLOR文やBGグラフィックスで色を変えると同時に色が変わってしまう。つまり、実線が色の境界線というわけだ。

実際のBGグラフィックスは、下3段には、図を描くことはできないが、プログラム中でPRINT文で出力することができる。そのため、参考に表には入れておいた。

コピーして、ゲーム作りに役立ててほしい。



できるということである。人間のイメージの世界では、なんの制限もなく自由な像・動きを生み出すことができる。しかし、いったんそれをコンピュータの画面にするとなると、メモリの制限、処理速度の制限、特にファミリーベーシックでは、キャラクターの制限なども生じてくる。画面で見たときに、いかにそれらしさを出すことができるかが、アイディアをゲームに生かすキーポイントとなってくる。

⑶全体を部品に分解してみよう

さて、アイディアが固まり、画面として形になったら、いよいよプログラムにとりかかる番だ。しかし、この段階でコンピュータのまえに腰をおろして頭からバリバリプログラムを組んでいくというわけには、なかなかいかない。そのためには、コンピュータの特徴をよく把握しているだけではなく、プログラムがどのように働き、どのように作られていかなければならないかをよく理解している必要がある。

まずすること。それは、これからプログラムしようとするゲーム全体の流れについて整理することである。最初に画面設定、点数のクリア、時間制限の設定などを行い、ゲームを始める下地を作る。つぎにコントローラーよりの入力と、パイプの切り替えを行う。そして、ジュースの発生、移動を行って、ボトルまでいきついたかどうか、そして、正しいボトルに入ったかどうかを判断する。ボトルにジュースがふえていく処理、ボトルがいっぱいになったら、次のボトルを用意する処理、違う色のジュースがボトルに入ってしまった時のミスの処理。そして、タイマーを動かして、制限時間を示す処理。最後に、時間切れで終了した時の得点の表示。

以上のように全体の流れを大まかな機能別の部品に分けることが、ゲームを能率よくプログラムしていくための重要なポイントである。

⑷部品を1コずつ作っていこう

全体の流れを、機能別の部品に分けたら、次はその部品のうちで、さらに細かい処理の順序を決めていく。そして、ベーシックの命令ひとつひとつでその細かい処理を置き換えていくのである。

全体としては複雑な動きを示すゲームプログラムも、部品に分けていけば、単純な動作の組み合わせでできているのだ。自分がプログラムできると思う所まで、機能を分けていき、問題を単純化していく。これが、プログラムを組む最良の方法であろう。

では、具体的に、「ジューススタンドゲーム」についての部品の内容を機能別に見ていくことにしよう。

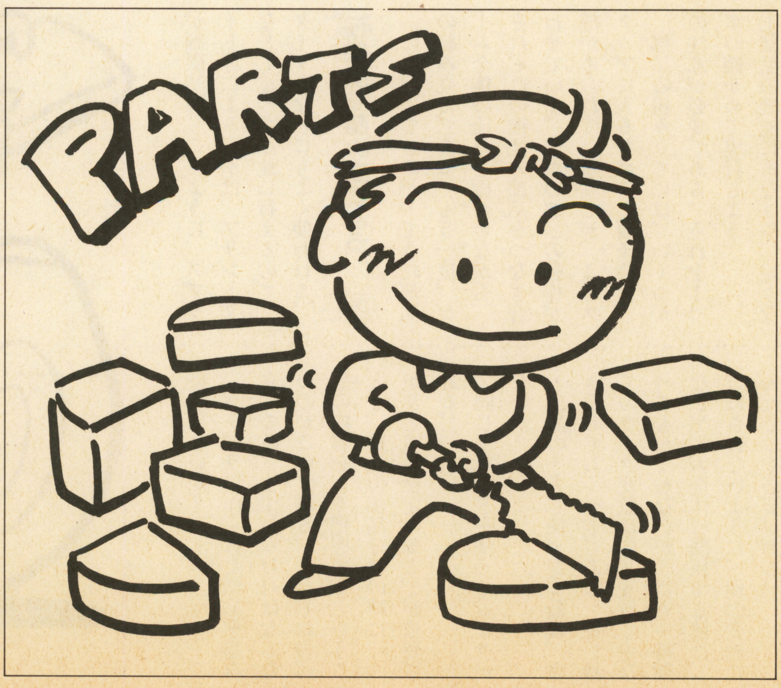
ⓐプレイヤーの入力

このゲームでは、プレイヤーはジュースの流れるパイプを、3つの分岐点で切り替えることができる。それによってジュースの流れる方向を変えるのである。

ゲーム性をもたせるため切り替えに少し制限を作った。切り替えは、分岐部1→2→3と順に行われ、Aボタンで切り替え、Bボタンでそのままという選択式とした。AまたはBボタンが押されると、次の分岐に選択が移るようになっている。その時に変更できる分岐点と、その状態は画面に示されるようになっている。

ⓑパイプの切り替え

パイプの分岐点は、上からやってきたジュースを、右、左、下の3方向に流れさせられるようにな





っている。ここでもゲーム性を出すために、分岐点の切り替えは、右→左→下の順に行われるようになっている。つまり、右方向からいきなり下方向へは切り替えられないということである。それには右をまず左に切り替えて、次に他の2つの分岐点についても選択を行った後に、左から下に切り替えるということになる。

1分岐に3つの選択があり、1回ずつしか状態を変えられないので、ジュースの動きを見ながら、すばやく切り替えることが必要となり、よりゲーム性を高めることができる。

ⓒジュースの動き

パイプを流れるジュースは、画面上方の3カ所から作り出される。そして、パイプをつたわって、流れていくわけだが、当然、横方向と縦方向では、移動するスピードが変わってくる。タイミングをとって切り替えることが必要になる。

ジュースの移動処理は、次にジュースが移動する所にあるパイプの状況で、移動方向、速度等が決定されるようになっている。パイプが╪で交差している所では、上から来たジュースは下に、右からは左に、左からは右に流れるようになっている。

ⓓボトルにつまるアニメーション

パイプの端まで来たジュースは、ボトルの中にポタリと落ちる。そしてジュースが少したまった形になる。この時、そのボトルに、すでにジュースが入っている時は、同じ色のジュースをそこに集めるようにしなくてはならない。違った色のジュースが混ざったら、そのボトルは失敗で捨てられてしまう。同じ色のジュースを一杯まで集めれば、1本終了で次のカラボトルがセットされる。ジュースはスプライトで書かれているが、ボトル内のジュースは、PRINT文で背景の面に書き込まれるようになっている。

ⓔタイマーの設定

このゲームは、一定時間内に、何本のジュースを作れるかというゲームなので、時間を制限する処理が必要となる。画面下の旗がそれにあたる。左端のSTARTから、右端のENDまでうごく間が制限時間。このように、スプライト等による表示は、プリント文などと違って高速に行えるので、タイマーなどにもよく使われる。

⑸部品を組み立てよう

いよいよできあがってきた部品の組み立てである。もう一度、ゲーム全体の流れを確かめながら、それぞれの機能の部品をつなぎ合わせていく。個々の部品の動作が確かめられていることが、このステップに入るための必要条件だ。

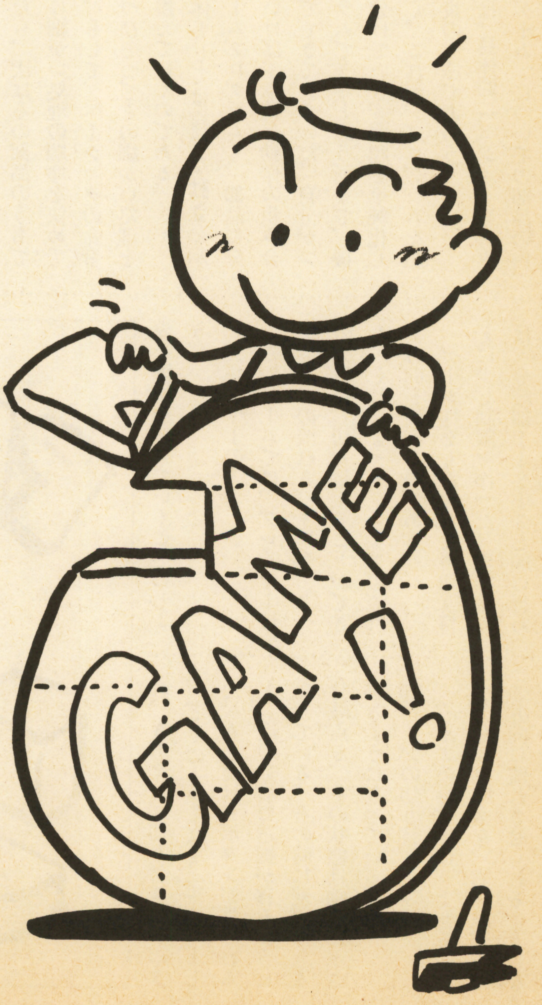
部品の組み立てには、いくつかの注意が必要である。それを1つずつ確かめて実行しよう。

①部品の機能に重なりや、足りない所はないか?

バラバラに作られた部品を組み立てる時、まず見なければならないのがこれである。それぞれの部品の内容をよく把握するということと、最初の部品分けの時にしっかりと境界を定めることが重要である。

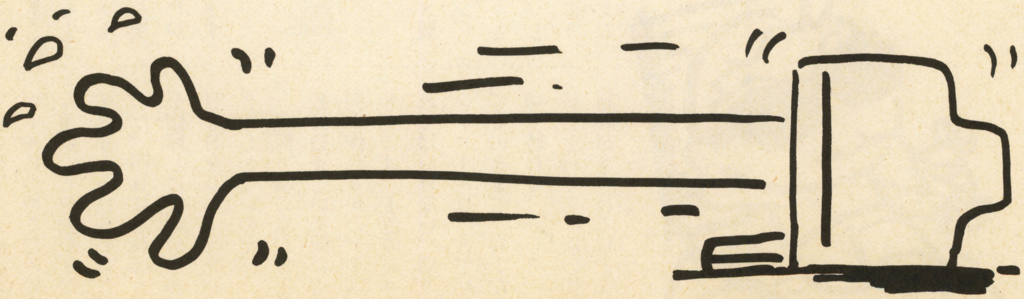
②同じ変数などを使って重なりがないか?

部品のうちで使った変数は、書き出しておくべきである。使い捨てのワーク変数は同じ変数で重なってもかまわないが、得点や時間などゲーム全体に関係する値は、こわされないように、別の変数に入れておく必要がある。



なみだのジューススタンド

```
10 Z=0:Y=1:X=2:I=Z:CGSETY,Y:SPRITEON:CGEN3:VIEW:U=32:PLAY"T1"
12 E=48:B=33:D=31:FORA=0TO3:READC,F:PALETS A,D,E,B,C:PALETB A,D,E,B,F:NEXT:DATA2
,2,22,2,39,22,39,39
15 DIM V(X),X(X),Y(X),J(X,X),B(Y,X),M(X),P(X)
18 FORA=Z TOX:READV(A),X(A),Y(A):B(Z,A)=999:NEXT:DATA1,7,6,0,12,9,2,16,7
20 LOCATE9,21:PRINT"ｼﾞｭｰｽ ｽﾀﾝﾄﾞ ｹﾞｰﾑ"
30 DEFSPRITE7,(Z,Z,Z,Z,Z)=CHR$(199):SPRITE7,16,200
40 LOCATE0,23:PRINT"START                         END";
50 FORA=Z TOX:DEFSPRITEA,(A,Z,Z,Z,Z)=CHR$(207):NEXT
90 C=Z
100 L=C:IFL>X THENL=L-3
102 GOTO1000
105 D=J(Z,L):E=J(Y,L):F=J(X,L):IFD=Z ANDRND(10)<5THEND=3:E=RND(3)*8:F=0
106 IFD=Z THEN 120
110 OND GOSUB150,160,170
115 IFF>19THEN500
120 C=C+Y:IFC>5THENC=Z:U=U+1:SPRITE7,U/2,200:IFU=464THENLOCATE10,10:PRINT"GAME O
VER":END
130 GOTO100
150 G=E+Y:H=F:GOTO200
160 G=E-Y:H=F:GOTO200
170 G=E:H=F+Y
200 M=ASC(SCR$(G,H)):N=D:IFM=221ORM=227THEN300
210 N=3:IFM=222ORM=223THEN300
220 N=2:IFM=225THEN300
230 N=1:IFM=224THEN300
240 N=3
300 IFD<>3ANDC>X THENRETURN
310 D=N:E=G:F=H:SPRITEL,E*8+16,F*8+23:J(Z,L)=D:J(Y,L)=E:J(X,L)=F:RETURN
500 SPRITEL:J(Z,L)=Z:A=(J(Y,L)-3)/6:IFB(Z,A)=999THENB(Z,A)=L:B(Y,A)=Z:GOTO520
510 IFB(Z,A)<>L THEN M(A)=M(A)+Y:LOCATE7*A+7,20:PRINTM(A);:GOTO600
520 B(Y,A)=B(Y,A)+Y:O=7*A+3:P=21-B(Y,A):COLORO,P,L+Y:COLORO+X,P,L+Y:LOCATEO,P:PR
INTCHR$(255);CHR$(255);CHR$(255);
530 IFB(Y,A)<3THEN120
590 P(A)=P(A)+Y:LOCATE7*A+7,18:PRINTP(A);
600 FORB=18TO19:O=A*7+3:COLORO,B,Z:COLORO+X,B,Z:LOCATEO,B:PRINTCHR$(244);CHR$(24
4);CHR$(244);:NEXT
610 LOCATEA*7+3,20:PRINTCHR$(245);CHR$(246);CHR$(248);
620 B(Z,A)=999:GOTO120
1000 J=STRIG(Z):IFJ=Z THEN105
1010 IFJ=4THEN1090
1020 IFJ<>8THEN105
1030 K=V(I)+Y:IFK>X THENK=Z
1040 V(I)=K:LOCATEX(I),Y(I):PRINTCHR$(224+K);:LOCATE25,4*I+5:PRINTCHR$(224+K);"
 ";:PLAY"O0C1D":GOTO1100
1090 LOCATE27,4*I+5:PRINT" ";
1100 I=I+Y:IFI>X THENI=Z
1110 LOCATE27,4*I+5:PRINTCHR$(196);:GOTO105
```

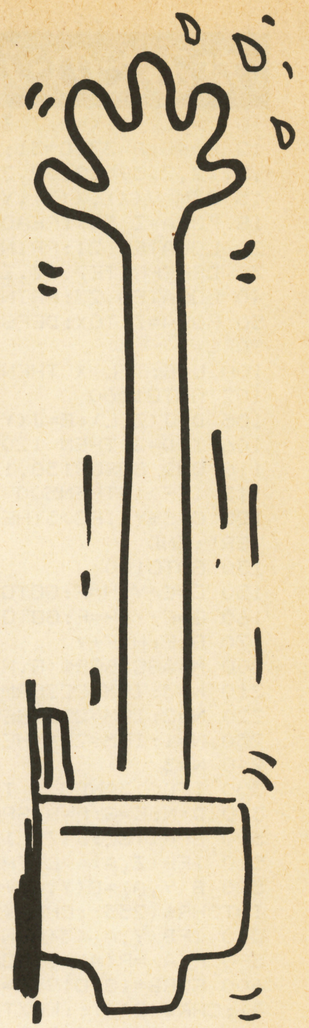


BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	D40								D40								D40											
1	J20								J20								J20											
2	J20				I60	J30	J30	J30	J10								J00	J30	I70									
3	J20				J20														J20									
4	J20				J00	J30	J30	I70											J20									
5	J20							J20											J20					I		J10		F40
6	J00	J30	J30	J30	J30	J30	J30	J10	J30	J30	J30	J30	J30	I70			J20		J20									
7								J20	I					J00	J30	J30	J20	J30	J10									
8								J00	J30	J30	J30	J30	I60	J30	J30	J30	J10	3										
9								I60	J30	J30	J30	J30	J00	J30	J30	J30	J30	J30	J30	I70				2		J00		
10								J20					J20	2						J20								
11								J00	J30	J30	J70		J20							J20								
12						I60	J30	J30	J30	J30	J50	J30	J10							J20								
13						J20					J20									J20				3		J20		
14						J20					J20									J20								
15						J20					J20									J20								
16																				J20								
17																												
18				L40	L40	L40	○		0		L40	L40	L40	○		0		L40	L40	L40	○		0					
19				L40	L40	L40					L40	L40	L40					L40	L40	L40								
20				L50	L60	M00	×		0		L50	L60	M00	×		0		L50	L60	M00	×		0					

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	D40								D40								D40											
1	J20								J20								J20											
2	J20							I60	J10								J20											
3	J20							J20									J20											
4	J00	J30	J30	J30	J30	J30	J30	I50	J30	J30	J30	J30	I70				J20											
5								J20					J20				J20							I		J10		F40
6			I60	J30	J30	J30	J30	J10	J30	J30	J30	I70	J20				J20											
7			J20					J20	I	I60	J30	I50	I50	J30	J30	J30	J20	J30	I70									
8			J20					J20		J20		J20	J20				J20	3	J20									
9			J20			I60	J30	I50	J30	I50	J30	I50	J00	J30	J30	J30	I50	J30	I50	J30	J30	I70		2		J00		
10			J20			J20		J20		J20		J20	J20	2			J20		J20			J20						
11			J00	J30	J30	I50	J30	I50	J30	I50	J30	I50	I50	J30	J30	J30	I50	J30	I50	I70		J20						
12						J20	I60	I50	J30	I50	J30	I50	I50	J30	J30	J30	I50	J30	J10	J20		J20						
13				I60	J30	J10	J20	J20		J20		J20	J20				J20	I60	J30	I50	J30	J10		3		J20		
14				J20		I60	J10	J20		J20	I60	I50	J10				J00	I50	J30	I50	I70							
15				J20	I60	I50	J30	J10		J20	J20	J20						J20	I60	J10	J20							
16				J20	J20	J20				J00	I50	I50	I70					J20	J20	I60	J10							
17																												
18				L40	L40	L40	○		0		L40	L40	L40	○		0		L40	L40	L40	○	0						
19				L40	L40	L40					L40	L40	L40					L40	L40	L40								
20				L50	L60	M00	×		0		L50	L60	M00	×		0		L50	L60	M00	×	0						



⑹ゲーム入力法

けっこう大きなプログラムなので、つめてリストどおりに打ちこんでほしい。カセットに必ずセーブしよう。BGグラフィックスも2面用意した。パイプの配置を変えたもので、多少ゲームの感じが変わってくると思う。背景もまずセーブしよう。

⑺遊び方

RUN↵でゲームスタート。

ここは、中小企業のジュース製造工場。いつもは全自動のコンピュータ制御(せいぎょ)でジュースを次々とボトルにつめて出荷している。しかし、プログラマーのミスによりバグが発生。コンピュータによるコントロールが不能となってしまった。3つの流出口から、いろんな色のジュースが流れだしてくる。ボトルにつめるための切り替え機も手動でしか動かせない。しかし、生産率を下げるわけにはいかないのだ。明日になればコンピュータを修理しに技師がやってくる。今日1日をヒューマンパワーでのりきるしかないのだ。

画面下端の旗がSTARTからENDまでいく間に、ジュースを何本作れるかのゲーム。1本のボトルには、1色のジュースしか入れてはいけない。切り替え機のコントロールは、Aで切り替え、Bでそのままで、右→左→下の順に切り替わる。そして、1つの切り替え機に替えを行うと、次の切り替え機に制御がうつってしまう。

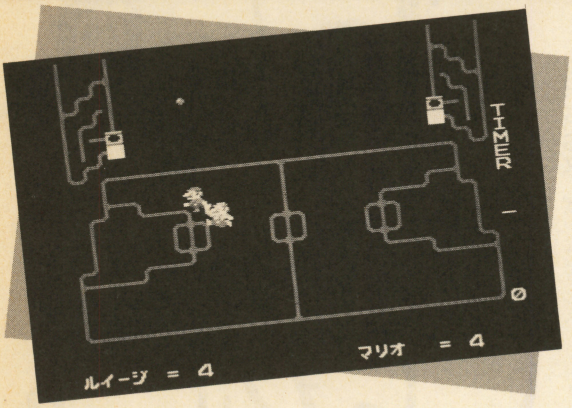


初級編

第5章

ファミリーベーシック秘技公開

ファミリーベーシック変わりダネ命令いろいろ



初級編は、この章でおしまい。ゲームを作っていくために、そのプログラム言語の特徴をつかんでおこう。たった2Kのメモリで、しかもベーシックである程度動きのあるゲームが作れてしまう。不思議なファミリーベーシックの特徴を、全体的に、また、代表的な命令別に見てみよう。

(1)ファミリーベーシックの特徴

まずは、ファミリーコンピュータがなぜこんなに人気があるか考えてみよう。

やはり、美しい画面とすばやい動きが大きな魅力だろう。これを生みだしているのは、ファミリーコンピュータのスプライト機能と、パレット機能だ。

画面の好きな位置にドットきざみに好きなキャラクターを置くことができるスプライト機能。一度設定すれば、あとは座標を指定するだけで高速で移動できる。しかもスプライトのキャラクターは、3色カラーが使用できる。重ね合わせの美しさも大きな特徴だ。この機能は、ベーシックでもそのまま使えるようになっている。SPRITE文とMOVE文がそれだ。これゆえ、ベーシックでも動きのあるゲームを作ることができるのだ。

もう1つは52色という表示色の豊かさだ。しかも、好みのキャラクターに色をつけるパレット機能がついている。ベーシックのCOLOR文とPALET文で制御(せいぎょ)できるようになっている。

(2)命令別特色

その他、ゲーム作りに必要な多くの命令が満載(まんさい)されたこのファミリーベーシック。特徴的命令について、簡単に説明してみよう。

⦿SPRITE文…スプライト番号0~7の8枚のスプライトを制御(せいぎょ)する命令。DEFSPRITE文で、次に示すようなスプライトの性質を指定し、SPRITE文で表示位置を制御(せいぎょ)する。

スプライト番号…0~7の番号を指定する。数字の小さいほうが、重なったとき、画面のより手前に表示される。

配色番号…0~3までの4組の色のコンビネーションより選択する。実際につける色は、PALET文で変更できる。

キャラクター合成型・8×8ドットのものと、16×16ドットの2種が選べる。

表示優先度…背景より前に表示するか、後ろにするかを決定する。

X軸・Y軸反転指示…左右・上下についてキャラクターを反転させて定義するのに用いる。

⦿MOVE文…キャラクターテーブルAのキャラクターを移動のアニメーションを使って動かしてくれる便利な命令。DEFMOVE文で後述の移動条件をセットして、MOVE、CUT、ERAで、移動、中断、消去させる。POSITIONやXPOS、YPOSなどで移動開始位置の指定や現在位置の確認などもできる。

アニメキャラクター動作番号・スプライト番号と同じに0~7を指定し、重なったときの表示の優先度を指定する。

アニメキャラクターの種類・16種のキャラクターのうち動かしたいキャラクターを選択する。

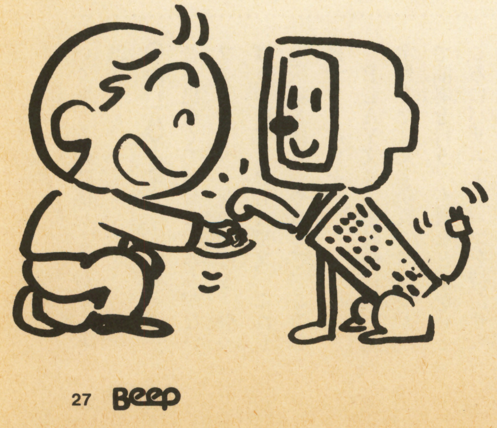
動作の方向…0で停止、1~8で上下左右斜めの移動方向を指定する。

移動速度…約1/30秒を最小単位として、1~255倍の時間に2ドットずつ移動するように指定する。

全移動量…0~255を指定して指定量の2倍ドット分移動させる。

表示優先度…表示を背景の前にするか後ろにするかを指定する。

配色番号…0~3の4種の配色番





号より色の組み合わせを選ぶ。

⊙CGEN文…ファミリーベーシックでは、扱うことのできるキャラクターを2つに分けて、アニメキャラクターのテーブルAと、文字と擬似(ぎじ)グラフィックのテーブルBとしている。BGグラフィックスで使えるのはテーブルBのほうだけで、だいたい背景にはテーブルBのキャラクターが使われ、スプライトにはテーブルAのキャラクターが使われる。しかし、CGEN文を使えば、背景とスプライトに、AまたはBのテーブルのキャラクターを指定することができる。これによって1画面内に多くのキャラクターをミックスさせることができる。

**⊙SCR$文**…ゲームなどで、キャラクターを移動させるときに必要となってくる処理に、衝突(しょうとつ)の判断というのがある。キャラクターがこれから動こうとするところに何があるかを知ることが必要となってくるわけだ。これを背景画面について行うのがSCR$である。キャラクターのX、Yの位置を指定して、その位置のキャラクターコードや配色番号を読み出せる関数だ。

⊙CGSET文…画面のキャラクターにつける色について、ファミリーベーシックでは、あらかじめ、キャラクターを見分けやすい色の組み合わせを何組か用意してある。それを指定するのがCGSET文。

背景用には、0、1の2種類が、スプライト用には0~2の3種類が用意されている。色については、マニュアルの最後にあるカラーチャートを参照してほしい。とりあえずCGSETで見分けのつく色にしておいて、後でPALET文で個々のキャラクターの色を調節してやればよいわけだ。

⊙PALET文…これが画面の色を直接決定する命令だ。詳(くわ)しいことは、上級編で再度説明するとして、ここでは軽く触れておくだけにしよう。

まず背景の色を変更するのか、スプライトの色を変更するのかをBまたはSの文字で指定する。次にカラーグループ番号0~3を指定する。これは、後述のCOLOR文やDEFSPRITE、DEFMOVEで指定した配色番号に相当するもので、その番号で指定されたキャラクターの色を変更することになる。

次にバックドロップの色(これは画面全体の色で、配色番号が0の時のみセットされる)、カラーチャートの左端、中央、右端の順に、4つの色コードを指定する。コードによる色の違いは、マニュアルのPALET文のところにかんたんに出ているが、上級編のパレットツールを使えば、実際に画面の上で見ることができる。

⊙COLOR文：背景画面を2×2の4文字ごとの区画に分けて、その区画内の配色番号を指定する命令。背景画面では、残念ながら1キャラクターごとに色を変えることはできない。X、Yの位置を指定して、その配色番号を指定してやると、その位置を含む2×2キャラクターのエリア内の4文字の色が同時に変わってしまう。背景画面の色を変えるときには、注意が必要だ。エリアの境界線は、X方向は画面左端から2文字おき、Y方向は上端・下端は1行ずつで、その間が2行おきとなっている。BGグラフィックスで、画面全体に絵を描いて、部分の色を変えていくと、その関係がよくわかると思う。

⊙PLAY文…ゲームはやっぱり音が出なくてはおもしろみがない。ファミリーベーシックにも、やっぱりついてるPLAY文。これも上級編で詳しくやるので、ここではサワリだけを。

PLAY文は、その後に続く文字列または文字変数の内容によって演奏(えんそう)を行う。音質など細かいところにバリエーションを持たせられるし、3チャンネルで和音を出したりもできるのだが、PLAY文実行中に、ベーシックのMOVE文以外が止まってしまうのが残念である。指定できるデータは、

・テンポ(T)…1~8で全体の速さを指定する。
・音色(Y)…0~3で音色を変えて、効果音を作ることができる。
・エンベロープ(M、V)…音がじょじょに消えていく効果を指定する。またエンベロープをかけないときは、音の大きさを指定する。
・オクターブ(O)…0~5で6オクターブの指定ができる。
・音程…C、D、E、F、G、A、Bと#で音程を示す。
・休符(R)…休符の指定。
・長さ…0~9の数字で32分音符から全音符までの音符、または休符の長さを指定する。

マリオとルイージのVSバスケット

```
100 Z=&H0:Q=&H1:R=&H2:S=&H4:U=Z:VIEW:LOCATEZ,23:PRINT"ﾙｲｰｼﾞ =";Z;:LOCATE17,23:PR
INT"ﾏﾘｵ  =";Z;:SPRITEON:FORA=Z TO&H4:DEFSPRITEA,(A ANDQ,Q,Z,A ANDQ,Z)="":NEXT:DE
FSPRITER,(Q,Z,Z,Z,Z)=CHR$(223):X=164:Y=120:T=&H4:V=Z:W=Z:K=Z:E=60:F=120:H=Z:I=S:
J=Z:M=Q
110 LOCATE12,Q:PRINT"     ";:LOCATE12,Q:POKE&H4015,&H17:POKE&H4000,&H41,&HE1:DEFS
PRITE&H3,(Z,Z,Z,Z,Z)=CHR$(220)
120 SPRITE&H3,232,N/&H8
1000 L=K AND&H3:A=STICK(L):B=STRIG(L):W=W-SGN(W):C=Z:D=Z:IFA=C THENU=Q
1010 IFA ANDQ THENC=R
1020 IFA ANDR THENC=-R
1030 IFA ANDS THEND=R
1040 IFA AND&H8THEND=-R
1050 IFSGN(C)=(V/R-Q)THENC=C*R
1060 IFG<>L THEN C=C+C/R:D=D*R
1070 A=&H680:X=X+C:Y=Y+D:IFX<24ORX>198THENX=X-C
1080 IFY<82ORY>176THENY=Y-D
1090 IF&H8>ABS(X-E)ANDABS(Y-F)<&H8THENX=X-C:Y=Y-D:IFL=G THENX=X+C:Y=Y+D:E=E+C
1100 IFL THEN1140
1110 IFY>F ANDK=Z THEN1140
1120 IFY<=F ANDK=S THEN1140
1130 K=S-K:DEFSPRITEK,(Z,Q,Z,SGN(V),Z)=""
1140 IF(B ANDS)ANDW=Z THENV=S-V:DEFSPRITEK,(L,Q,Z,SGN(V),Z)="":W=&HA
1160 U=(U+Q)AND&H3:ONU+V+Q GOTO1170,1180,1190,1180,1200,1210,1220,1210
1170 POKEA+S*K,Z,Q,R,&H3:GOTO1230
1180 POKEA+S*K,S,&H5,&H6,&H7:GOTO1230
1190 POKEA+S*K,&H8,&H9,&HA,&HB:GOTO1230
1200 POKEA+S*K,Q,Z,&H3,R:GOTO1230
1210 POKEA+S*K,&H5,S,&H7,&H6:GOTO1230
1220 POKEA+S*K,&H9,&H8,&HB,&HA
1230 T=S-T:IFT THENPOKE&H4015,&H17:POKE&H4011,&H1F
1240 IFG=L THENSPRITER,X+V*V,Y+T:GOTO1320
1250 SPRITER,E+I*I,F+T:IFW THEN1320
1260 IFV+I<>S THEN1290
1270 IFABS(Y-F)>S ORABS(X-E)>12THEN1320
1280 GOTO1300
1290 IFABS(X-E)>S ORABS(Y-F)>12THEN1320
1300 IFI/R-Q<>SGN(X-E)THEN1320
1310 G=Q-G:J=&HA:POKE&H4002,&H80,&H8
1320 SPRITES-K:IF(B=&H8)AND(SGN(V)=L)AND(G=L)THEN2000
1330 SPRITEK,X+S,Y-&H6:SWAPX,E:SWAPY,F:SWAPU,H:SWAPV,I:SWAPJ,W:SWAPK,M:N=N+Q:IFN
<1520THEN120
1910 LOCATE9,Q:PRINT"GAME OVER";:IFSTRIG(Z)<>Q THEN1910
1920 RUN
2000 IFL THENPOKEA+S*K,&H5,S,&H7,&H6:GOTO2020
2010 POKEA+S*K,S,&H5,&H6,&H7
2020 FORB=Y TOY-&H16STEP-R:SPRITEK,X+S,B-&H6:SPRITER,X+V*V,B:PAUSEQ:NEXT:IFL THE
NPOKEA+S*K,&HD,&HC,&HF,&HE:GOTO2040
2030 POKEA+S*K,&HC,&HD,&HE,&HF
2040 FORB=Y-&H16TOY STEPR:SPRITEK,X+S,B-&H6:PAUSEQ:NEXT:B=L*160+40:G=X+V*V:H=Y-&
H16:F=(B+G)/R
2050 J=SGN(F-G):C=(H-&HA)/&H5:G=G+J:H=H-C:SPRITER,G,H:PAUSEQ:IFC THEN2050
2060 J=SGN(B-G):C=(H-14)/&H5+Q:G=G+J:H=H+C:SPRITER,G,H:PAUSEQ:IFH>Y THEN2100
2070 IFH<55ORH>65THEN2060
2080 IFABS(B-G)<=S THENPRINT"GOAL":GOTO2110
2090 GOTO2060
2100 POKE &H4000,&HF,&HF4,&HFD,&H8:PRINT"MISS":GOTO2130
2110 POKE &H4000,&H4F,&HCB,&HFD,&H8:IFL THENO=O+R:LOCATE&H6,23:PRINTO;:GOTO2130
2120 P=P+R:LOCATE23,23:PRINTP;
2130 E=B-&H8:F=120:H=Z:J=Z:PAUSE150:G=Q-L:W=&HA:GOTO110
```



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	K60			K60						V	S		B	A	S	K	E	T						K60			K60	
1	K60		K70	K30																				K40	L00		K60	
2	K60	K70	L20	K60																				K60	L10	L00	K60	
3	K40	L20	K70	K60																				K60	L00	L10	K30	
4	K60	K70	L20	K60																				K60	L10	L00	K60	
5	K60	K60	K50	H10																				H10	K50	K60	K60	
6	K60	K60	K70	M50																				M50	L00	K60	K60	T
7	K60	K70	L20																						L10	L00	K60	I
8	L10	L20	K70	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K20	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	L00	L10	L20	M
9			K60											K60											K60			E
10			K40	K50	L00									K60									K70	K50	K30			R
11		K70	L20		L10	K50	K50	L00						K60						K70	K50	K50	L20		L10	L00		
12		K60					K70	K00	L00				K70	K00	L00				K70	K00	L00					K60		
13		K60					K60	K60	K60				K60	K60	K60				K60	K60	K60					K60		
14		K60					L10	K00	L20				L10	K00	L20				L10	K00	L20					K60		
15	K70	L20			K70	K50	K50	L20						K60						L10	K50	K50	L00			L10	L00	
16	K40	K50	K50	K50	L20									K60									L10	K50	K50	K50	K30	
17	K60													K60													K60	
18	K60													K60													K60	
19	K60													K60													K60	
20	L10	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K10	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	L20	0

・重音…A、B、Cの3チャンネルを、・・(コロン)で区切って、続けると、同時に3音出すことができる。

⊙STICK文…コントローラーの✜ボタンからの入力を値として返す関数。右1、左2、下4、上8の値を返してくる。ナナメが押されたときは、その値の和が値として返される。右ナナメ下ならば1＋4＝5である。何も押されていない場合には0が返されてくる。

0でコントローラー1、1でコントローラー2が読み出せる。

⊙STRIG…コントローラーのスイッチA、B、SELECT、それに、START を読み出す関数。0ではコントローラー1で、START1、SELECT2、B4、A8を値として返し、1ではコントローラー2のB4、A8を値として返してくる。2つ以上押されたときは、それぞれの値の和の値が返されてくる。

(3)入力法

またまた、メモリいっぱいのプログラム。リストどおりに打ちこんでほしい。余分なスペースなど入れないように。BGグラフィックスで背景を入れたら、プログラムも背景もカセットにセーブしておこう。暴走して、せっかくの努力が水の泡(あわ)とならないために。

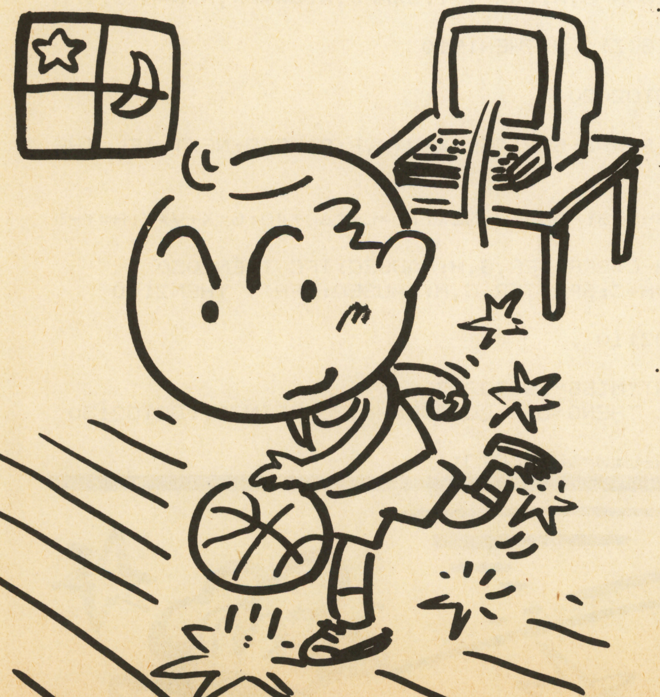
(4)遊び方

このゲームは、2人で対決するバスケットボールもどきゲームである。

RUN↵でゲームスタート。画面右端のタイマーが0になるまでの間に、相手のゴールにより多くのボールを投げこんだほうが勝ちだ。プレイヤー1は左のゴールに、プレイヤー2は右のゴールにシュートすれば得点となる。

移動はコントローラーの✜ボタンで行い、上下左右ナナメに動ける。Aボタンでシュート。シュートできるのは相手ゴールのほうを向いているときのみ。使い方は2とおり、相手ゴールの近くまでいき、ゴール目ざしてシュートするのと、自分のゴールの近くでは、ボールをクリアして相手をさがらせることができる。Bボタンは、右向き、左向きを切りかえるのに使う。

相手よりボールに近づけばボールを奪うことができ、ボールを奪われたり、Bボタンを押したりすると、ボール10回バウンドする間は、ふり向いたりボールを取ったりできないようになっている。



中級編

実技をたくさんとり入れて、ファミリーベーシックを使いこなそう

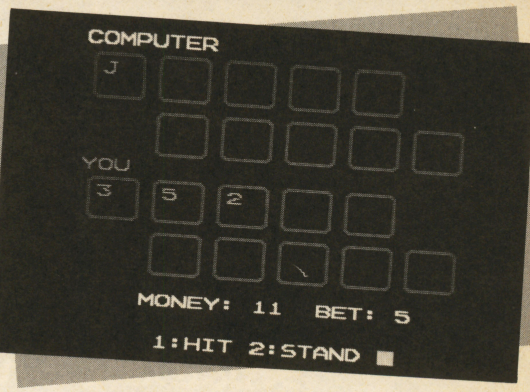


第1章

頭スッキリ！フローチャート

フローチャートを考えてみよう

ゲームプログラムを楽々作りこなすためには、まずゲーム作りのものの考え方、すすめ方を身につけてしまうのが一番の早道。
このフローチャートさえあれば君の頭のなかは、いつもクリーンに整理され、プログラム全体の流れがバッチリ頭の中に描かれてしまうというすぐれもの。これで巨大なプログラムにも大挑戦！



(1)フローチャートって何だ

ゲームプログラムを作るにあたってまず考えなければならないのは、そのプログラムがどのような要素から構成され、どう結びついているかということだ。

例をあげよう。そう、プラモデルの飛行機。これは買ってきたばかりのときは、バラバラな状態だ。ところが、バラバラになっている部品を正しく組み合わせていくと、本体や翼(つばさ)になり、やがては1台の飛行機となる。ただこのとき、余計な部品を使ったり、誤った組み合わせ方をすると飛行機は完成しない。つまり、適切な部品を選び出し、正しく組み合わせなければならないわけだ。

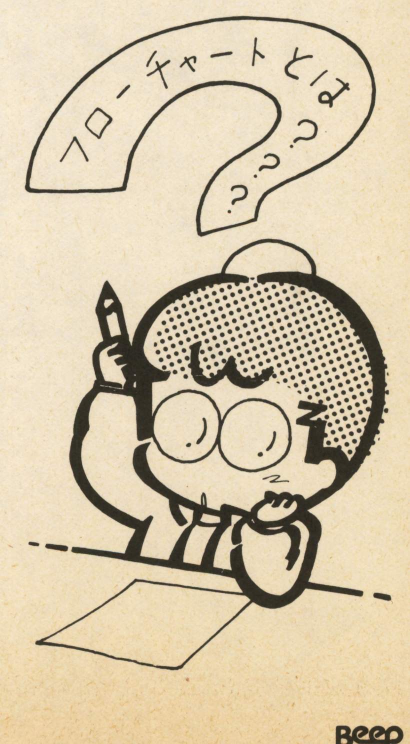
プログラムについても、まったく同じことがいえる。だからプログラム作成の前に必要な物は何で、それらをどう組み合わせたらよいかを考えてやらないといけない。

それでは具体的にどうすればいいのだろう。いろいろな方法があるが、その中にフローチャートというものがある。これは、プログラムのアルゴリズムを図式化したもので、流れ図ともいわれる。

図1と図2を見てほしい。これがフローチャートだ。このフローチャートの利点は、初心者にもわかりやすく、コーディングやデバッグがしやすいという点だ。もちろん欠点もある。巨大なプログラムや、条件分岐(ぶんき)の多いプログラムでは複雑になり、わかりにくいし、繰り返しが表現しにくい。

でも、入門には一番(いちばん)いいと思うし、知っておいても損にはならないので、本章ではこのフローチャートをとりあげることにした。フローチャートについて学ぶことで、論理的考え方を身につけようというのが目的だ。フローチャートの考え方から、書き方・実際例までバッチリ説明しちゃうから、この章を読みおえるころには、プログラマーの考え方がわかるようになっていることうけあいだ。しっかり読んでくれたまえ。

なお、実際の例として、トランプゲームの「21」のファミコン版、「21(ブラックジャック)」を使う。これからこのプログラムを元に説明していくが、その前に一度、理解しやすいように、実際にこのゲームをやってみてほしい。



(2)フローチャートの書き方

それではフローチャートの基本事項を説明しよう。まず見方だが、フローチャートは上から下に、左から右に見るというのが原則だ。もちろん例外もあるが、その場合は指示があるので、それに従う。

次はお待ちかねのフローチャートの書き方だ。これは正式には、『情報処理用流れ図記号(JIS・C・6270)』という規格になっている。ただこれは、正式な場合はこれに従いなさいということだ

21(ファミコン風ブラックジャック)

```
40 GD=10:DIMCD(1,10),AN(1),CT(1),CN(1)
1000 CLS:VIEW:GOSUB8000
1010 LOCATE5,22:INPUT"イクラ カケマスカ";BT:IF BT=0 THEN1820
1020 IF BT>GD THEN BEEP:GOTO1010
1030 GD=GD-BT:GOSUB8000
1100 CA=1:FORI=0TO1:AN(I)=0:CT(I)=0:NEXT
1110 FL=1:GOSUB3000:GOSUB3000:FL=0:GOSUB3000
1130 FL=1:BN=21:GOSUB5000:IF RS=2 THEN1600
1200 LOCATE5,22:PRINT"1:HIT 2:STAND ";:KI$=INKEY$(0)
1210 IF KI$="1" THEN1300
1220 IF KI$="2" THEN1400
1225 GOTO1200
1300 FL=1:GOSUB3000:BN=21:GOSUB5000:IF RS=1 THEN1700
1310 GOTO1200
1400 FL=0:GOSUB3000:BN=21:GOSUB5000:IF RS=1 THEN1600
1410 IF CN(0)<17 THEN1400
1500 IF CN(0)>=CN(1) THEN1700
1600 FL=1:GOSUB6000:GD=GD+BT*MU:GOSUB7000:GOTO1800
1700 FL=0:GOSUB7000
1800 IF GD>0 THEN1000
1820 CLS:LOCATE8,12:PRINT"GAME OVER":END
3000 '
3010 CA=CA+1:CT(FL)=CT(FL)+1:CX=RND(13):IF CX=0 THEN AN(FL)=AN(FL)+1
3020 CD(FL,CT(FL))=CX:GOSUB4000:RETURN
4000 '
4010 RESTORE4910:FORI=0TOCX:READD$:NEXT
4020 X=1+(CT(FL)-1)*4:Y=2+FL*9:IF CT(FL)>5 THEN X=X-16:Y=Y+4
4030 LOCATEX,Y:PRINTD$
4040 RETURN
4910 DATA A,2,3,4,5,6,7,8,9,10,J,Q,K
5000 '
5010 RS=0:CN(FL)=0:FOR I=1 TO CT(FL):X=CD(FL,I):IF X>8 THEN X=9
5020 IF X=0 THEN X=10
5030 CN(FL)=CN(FL)+X+1:NEXT
5040 IF CN(FL)=BN THEN RS=2:RETURN
5050 IF CN(FL)<BN THEN RETURN
5060 IF AN(FL)=0 THEN RS=1:RETURN
5070 FOR I=AN(FL) TO 1 STEP -1:CN(FL)=CN(FL)-10:IF CN(FL)<=BN THEN I=1
5080 NEXT:IF CN(FL)=BN THEN RS=2
5090 IF CN(FL)>BN THEN RS=1
5100 RETURN
6000 '
6010 M$="":MU=2:BN=21:GOSUB5000:IF ((CT(FL)=2) AND (RS=2)) THEN MU=4:M$="BLCK JA
CK":MU=6:GOTO6500
6020 J=0:FORI=0TO2:IF CD(FL,I)=6 THEN J=J+1
6030 NEXT:IF J=3 THEN M$="7-7-7":MU=4:GOTO6500
6040 J=5:FORI=0TO2:IF CD(FL,I)=J THEN J=J+1
6050 NEXT:IF J=8 THEN M$="6-7-8":MU=9:GOTO6500
6060 IF RS=2 THEN M$="21":MU=3
6070 FORI=5TO10:IF CT(FL)=I THEN M$=M$+STR$(I)+"マイ":MU=MU*(I-1):I=10
6080 NEXT
6500 LOCATE5,20:PRINTM$:RETURN
7000 M$="COMPUTER":IF FL=1 THEN M$="YOU"
7010 BT=0:GOSUB8000:LOCATE5,22:PRINTM$;" WIN !!        "
7025 FORI=0TO300
7030 KI$=INKEY$:IF KI$=" " THEN I=300
7035 NEXT
7040 RETURN
8000 LOCATE10,19:PRINTGD;" ":LOCATE19,19:PRINTBT;" ":RETURN
9000 'PROGRAM  VERSION 1.4
9010 'PROGRAMED BY T.UKITA
```

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	C	O	M	P	U	T	E	R																				
1	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02								
2	K62			K62	K62			K62	K62			K62	K62			K62	K62			K62								
3	K62			K62	K62			K62	K62			K62	K62			K62	K62			K62								
4	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22								
5					K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02				
6					K62			K62	K62			K62	K62			K62	K62			K62	K62			K62				
7					K62			K62	K62			K62	K62			K62	K62			K62	K62			K62				
8					L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22				
9	Y	O	U																									
10	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02								
11	K62			K62	K62			K62	K62			K62	K62			K62	K62			K62								
12	K62			K62	K62			K62	K62			K62	K62			K62	K62			K62								
13	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22								
14					K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02				
15					K62			K62	K62			K62	K62			K62	K62			K62	K62			K62				
16					K62			K62	K62			K62	K62			K62	K62			K62	K62			K62				
17					L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22				
18																												
19					M	O	N	E	Y	:						B	E	T	:									
20																												

から、自分だけのために使うときはそれほど気にすることはない。準じていればいい。なお、書き方は読み方と同様に、上から下に、左から右にというのが原則だ。これに従わないときは矢印で示してやらなければならない。

よく使われる記号は図3のようなものだ。これらの記号を並べて、線(流れ線という)で結べばいい。そして各記号のなかにそこで何を行うのか書きこんでやれば、いっちょうあがりだ。基本はこれだけ。かんたんなものさ。

さて、書き方がわかったところで、さっそく練習してみよう。今、次のようなプログラムを作るとする。『変数Aに変数Bの値を加え、変数Cに入れる。このCの値が10を超えたら、プログラムの実行を終了し、超えていなければ、Aに1を加えてから、Bの値を加える処理に戻る。なお、初期値はAが0、Bが1とする』このプログラムのフローチャートを書いてみよう。

一番最初は、図3のなかの端子という記号を使う。なかには「START」とか「始め」と書けばいい。次は初期設定の部分。これはプログラムを確実に思ったとおりに動かすのに必要不可欠なものだ。

さて、この初期設定部分は処理に相当するので、これを表す記号を使う。ここまで書くと図4のようになる。その次はメイン部分だ。A+Bの結果をCに入れた後、Cが10を超えたかどうか判定する。これを書きこむと、図5のようになる。あとは判定で分かれた後の処理だけだ。10を超えた場合とそうでない場合にどうすればいいのか考えて、書きこもう。

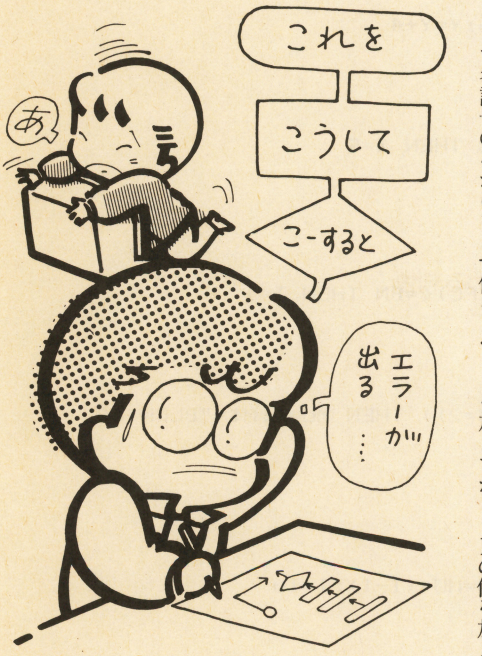
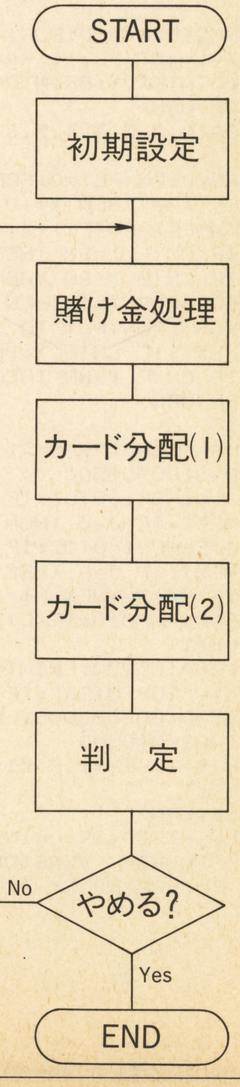


図1 ジェネラルフローチャート



処理はこれですべてだから、最後に線で各記号を結んで流れを示してやれば完成だ(図6)。キミの書いたものと比べてみよう。こんなふうになったかな。少し違ったかもしれないが、根本的に間違っていなければ構わない。プログラムの流れを視覚的にとらえることができるようになっていればOKだ。

(3)実際のフローチャートの使われ方

さて、読み方・書き方がわかったら、「21」のほうを考えてみることにしよう。トランプゲーム、「21」のゲームを作るということは、このルールをプログラム化するということである。そこでまず、実際のゲームの流れを考えてみる。

ゲームは、カードのシャッフル(初期化)から始まる。つぎは賭けだ。賭け終了後、カードが子と親に配られる。子はもう1枚ひくときにはHITを、いらない場合はSTANDを宣言する。子がつぶれた場合は自動的に親の勝ちとなり、ゲーム終了となるが、STANDした場合は、ゲームは続行となる。この場合、親はカードを開き、和が17以上になるまでカードをひく。つぶれなければ勝負となる。そしてその結果に基づいて、賭け金が移動する。以上が「21」の1サイクルである。

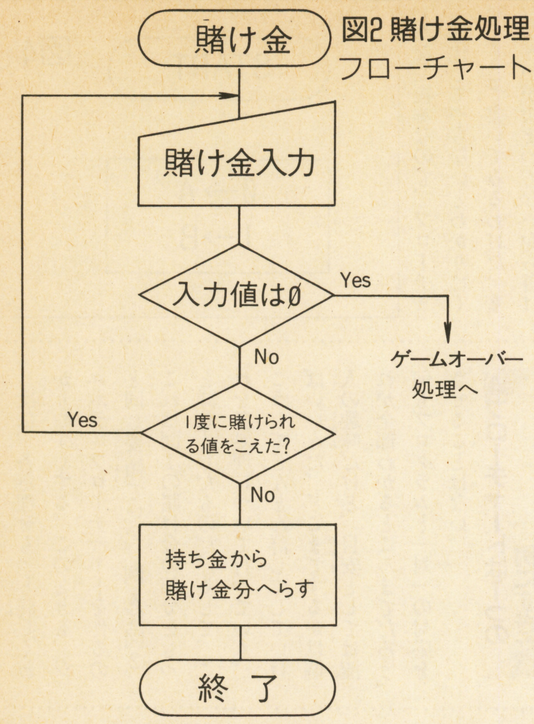


図2 賭け金処理フローチャート

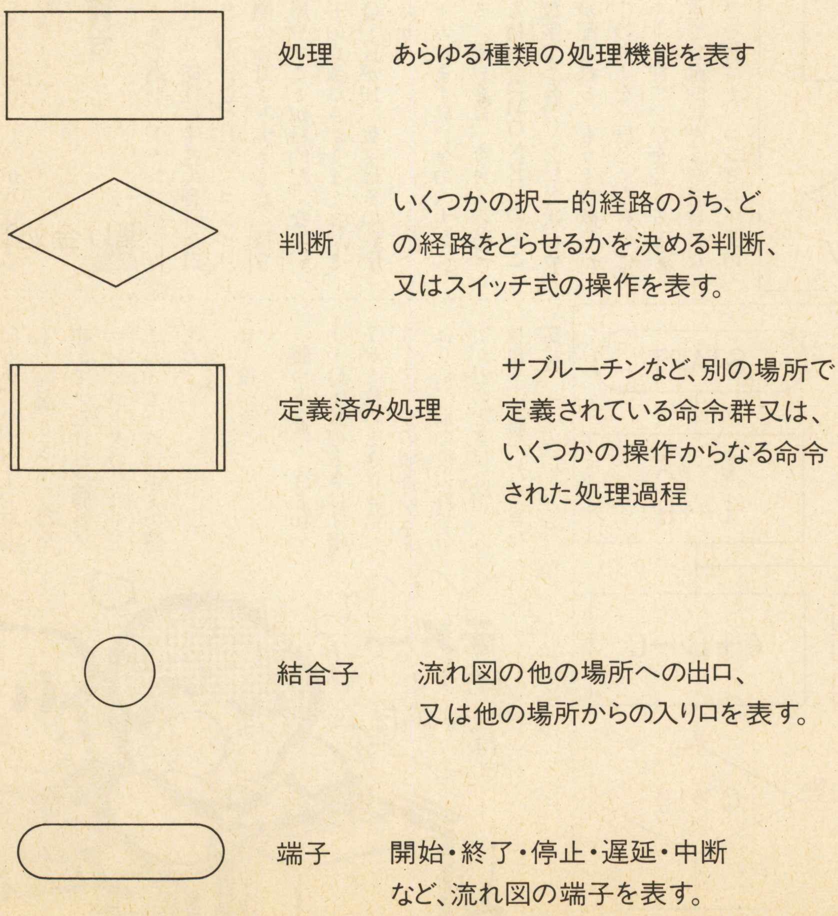
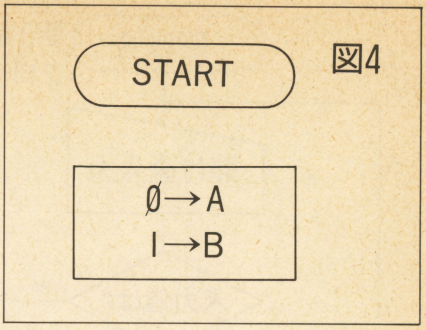


図3 フローチャート記号(一部)



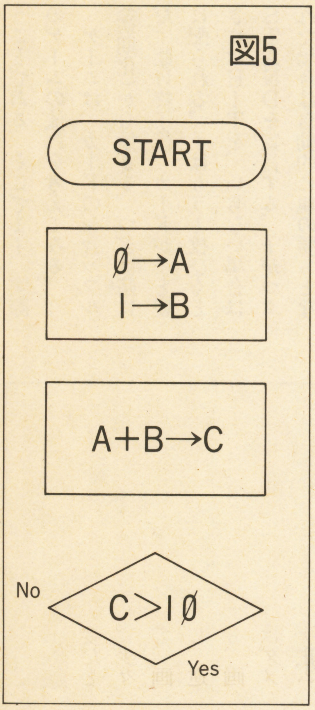
かなり複雑なので、フローチャート化してやらないとわかりにくい。そこで書いてみるわけだ。変数なんかが決まってないから書けないよっていう人がいるかもしれない。確かに決まっていない。でも書けるんだ。図1を見てほしい。えーっ、こんなのありかよっていうかな。でも、こうすればわかりやすいよね。こういうプログラム全体の流れをみるためのフローチャートを、ジェネラルフローチャートって言うんだ。先ほどのゲームの1サイクルを見事に示しているよね。これがあるのとないのでは、ゲーム作りの大変さが全然違う。

ゲームプログラム製作の第一歩は、このジェネラルフローチャート作成なんだ。とにかく、これを作ること。すべてはそこから始まると言っても言い過ぎでない。

次は、各処理をもっと詳しくわかりやすくする。このときも、ジェネラルフローチャートを作るのと同じ要領でいい。例えば図2。これは図1の賭け金の部分をもっと詳しくしたものだ。そして、これを各処理について行っていく。こうしてどんどん詳しくしていけばいいんだ。どこまでするかは個人の趣味だけど、目安としては流れが全部わかるところまでだろう。何もプログラムと一対一の対応をさせることはない。

(4) フローチャートは1つの例にすぎない

えんえんとフローチャートについて書いてきたが、フローチャートの役割と重要性はわかってもらえたかな。ポイントは自分で考えて書いてみること。そうすれば、おのずとプログラムが見えてくるハズだ。とにかく書いてみよう。そして一日も早く身につけてしまおう。学問に王道なしっていうことわざがあるけど、それはそっくりプログラム作りにもあてはまっちゃうんだよね。しっかり覚えておこう。



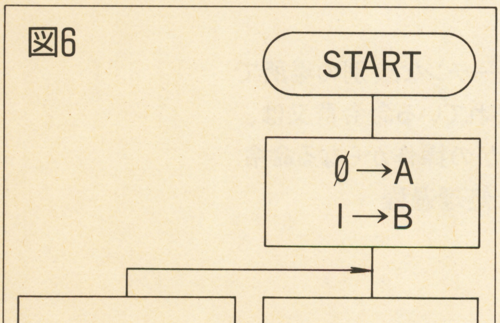
(5) 遊び方

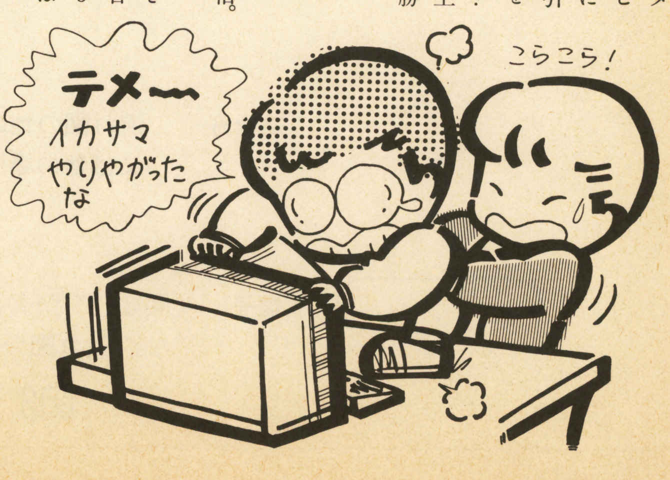
ひとりで遊び、トランプゲームの「21」と同様に手札の合計を21にできるだけ近づけて、コンピュータ側の合計より大きくし、持ち金を増やすことが目的だ。なお、カードの点数はAが1または11で、J・Q・Kが10、数字はその数字のとおり。

ゲームはまず賭け金の入力から始まる。賭け金は、今キミの持っている持ち金(MONEYのところの数字)まで賭けられる。次にカードが配られる。もう1枚欲しければ1のキーを、欲しくなければ2のキーを押す。21を超えるとその時点で自動的にキミの負けになる。2を押すと、コンピュータの番になり、コンピュータがカードを引く。コンピュータは合計が17以上になるまで必ずカードを引く。両者ともに、21を超えていなければ勝負！コンピュータの合計を上まわっていればキミの勝ちだ。

☆ 役 ☆

勝つと2倍。Aと10・J・Q・Kのときは4倍。7が3枚のときは4倍。6―7―8の順番のときは9倍。前の役以外で合計が21のときは3倍。5枚以上で勝ったときは(枚数―(マイナス)1)倍になる。

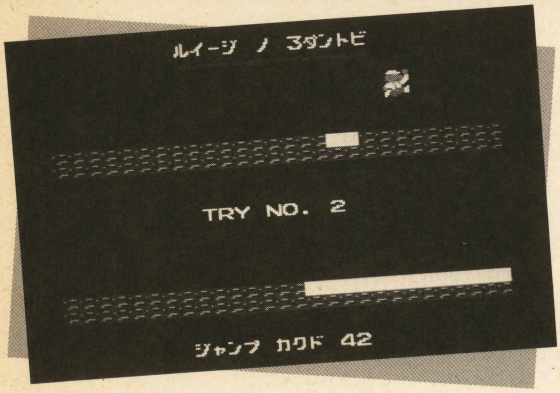




中級編

第2章 美人な画面づくりに挑戦！

いよいよキャラクターの登場



ゲームをより楽しくする方法その1。それは、画面をより美しくしてあげること。美人の画面を相手にすれば自然に心もウキウキしてくる。
そう！きれいで楽しい背景と魅力的なキャラクター作りこそが、人々を熱狂させてやまない、大ヒットゲームの欠かすことのできない条件なのだ。

⑴画面は重要だ

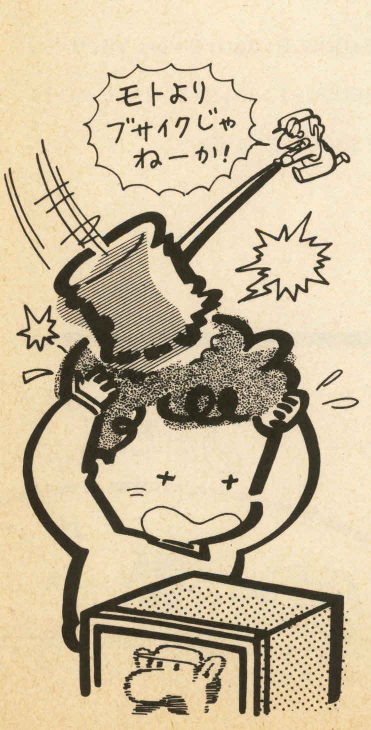
ゲームにとって画面が重要だということに文句はないよね。少々プログラムに難があったって、画面がきれいなら許しちゃうし、逆にどんなによくできていても、画面がいまいちなら、こいつはダメだってことになる。つまりゲームではブスより美人の方がいいってわけだ。だからせいぜい美人にしてあげよう。とすると、どうすれば画面を美人にできるのかってことになるよね。ところがどっこい、こいつは難しい問題だ。名付けて画面美人化問題だ。じっくりと取り組むことにしよう。

まずゲーム画面を構成するものを考えてみよう。いったい何と何からできているのだろうか。ずばり言ってしまえばキャラクターと背景ということになるだろう。キャラクターとは、マリオやレディのような動作をする物、背景とはこれらキャラクターが活躍する場のことを指す。だから画面美人化問題はこのキャラクターと背景の2つに分けて考えればいい。

⑵キャラクター登場！

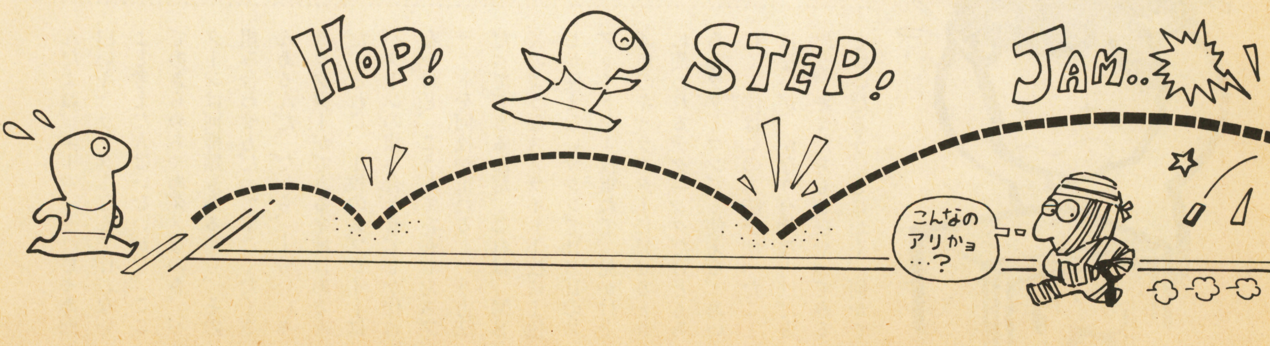
そこでまずキャラクターから考えてみよう。ヒットしたアーケードゲームのキャラクターを見てみると、次のような傾向が見られる。いわゆるシューティングタイプのゲーム（ゼビウスなど）ではリアルで質感のあるキャラクターが、そうでないものには実に可愛らしいキャラクターが多いということだ。つまり、ゲームの内容に沿った魅力的なキャラクターが登場しているわけだ。面白いといわれるゲームはキャラクターに魅力があるんだね。もちろんキャラクターは実に魅力的なんだけど面白くないというゲームも存在する。けれどもキャラクターがいいということは、やっぱりゲームを面白くするポイントの1つだ。

だから画面美人化問題の1つの答えは、いいキャラクターを作るということ。ところが、ファミリーベーシックの場合はスプライトパターンを自由に定義できず、既存のパターンを組み合わせてキャラクターを作らなければならないんだよね。エーッ、それじゃあ面白いキャラクターなんて作れないよと考えた人はいないかな。ここであきらめちゃあダメなのさ。パターンは何種類もあるから、その組み合わせ方だってすごくたくさんあるんだ。だから、十分いいキャラクターができる余地はあるんだ。それに、1枚のスプライトはパターンの組み合わせでしか作れないけど、こうして定義したスプライトを何枚か重ねるという方法だってできるんだ。これは、MSXのようなスプライト1枚に1色しか使えないマシンで16×16のスプライトを4枚重ねて、見かけ上4色のキャラクターに見せるときに使われるテクニックだけどもちろんファミコンでも使える。あいにくと僕はこの方法を使ったことはないが、かなり有効な方法だと思う。キミたちもぜひやってみるといいだろう。とにかく、何かで



ルイージの三段とび

```
1000 A=&H0:B=&H1:E=&H2:G=10:DIMR(E):CGSETB,A:PALETS A,15,48,38,18:FORI=A TOE:DEF
SPRITEI,(B,B,A,B,A)="":NEXT:POKE&H680,B,A,&H3,E,&H5,&H4,&H7,&H6,13,12,15,14:SPRI
TEON
1010 T=A
1020 VIEW:LOCATE16,12:PRINTT+B:X=16:Y=63:D=B:P=&H6:C=A:K=A:F=A:SPRITED,X,Y:PLAY"
T1Y102F3R1F9":POKE&H4015,&H17:POKE&H4000,E
1030 IFINKEY$=""THEN1030
1040 POKE&H4001,&H78,A,A:C=C+B:IFC>&H4 ANDP>A THENP=P-B:C=A
1050 W=ASC(INKEY$):IFW=32THEN1110
1060 IFW=&H1BTHEN1220
1070 IF(W=28ANDK=A)OR(W=A ANDK=28)THENC=A:K=W:P=P+B
1080 X=X+P/G:IFX>162THEN1190
1090 SPRITED:D=D+B:IFD=E THEND=A
1100 SPRITED,X,Y:POKE&H4001,&H78,&H80,&H9:GOTO1040
1110 GOSUB1300:P=P*11/G:GOSUB1300:P=P*11/G:GOSUB1300:IF(F=A ANDX<162)THEN1190
1120 LOCATE&H5,12:PRINT"ﾀﾀﾞｲﾏ ﾉ ｷﾛｸ      ":R=(X+51)*15/E:I=230:J=X:IFF=A THENR=(X
-163)*15/E:I=X:J=18
1130 DEFMOVE(A)=SPRITE(&H4,&H3,E,(I-162)/E-B,A,A):DEFMOVE(B)=SPRITE(&H4,&H3,E,(J
-16)/E-B,A,A):POSITIONA,163,63:POSITIONB,16,151:MOVEA
1140 W=(XPOS(A)-163)*15/E:GOSUB1280:IFMOVE(A)=-B THEN1140
1150 IFF=A THEN1180
1160 ERAA:MOVEB
1170 W=(XPOS(B)+51)*15/E:GOSUB1280:IFMOVE(B)=-B THEN1170
1180 W=R:GOSUB1290:LOCATE16,12:PRINTW$:PAUSE150:GOTO1200
1190 R=A:LOCATE11,&H3:PRINT"FOUL!  ":PLAY"T1Y202E5FGAEDC":PAUSE100
1200 GOSUB1270:R(T)=R:IFM<R THENM=R
1210 T=T+B:IFT<&H3THEN1020
1220 GOSUB1270:CLS:LOCATE&H7,&H3:PRINT"TODAY'S BEST":W=M:GOSUB1290:LOCATE&H9,&H5
:PRINTW$:LOCATE&H7,&H7:PRINT"YOUR RECORDS":FORI=A TOE:W=R(I):GOSUB1290:LOCATE&H7
,&H9+I*E:PRINTI+B"ｶｲﾒ:"W$:NEXT:LOCATE&H7,22:PRINT"REPLAY? (Y/N)"
1230 W$=INKEY$:IFW$=""THEN1230
1240 IFW$="Y"THENFORI=A TOE:R(I)=A:NEXT:GOTO1010
1250 IFW$<>"N"THEN1230
1260 CLS:END
1270 FORI=A TOE:SPRITEI:POSITIONI,A,A:ERAI:NEXT:RETURN
1280 PLAY"T1Y004":GOSUB1290:LOCATE16,12:PRINTW$:PLAY"E0G":RETURN
1290 W$=RIGHT$(STR$(W/100),E)+"M "+RIGHT$(STR$(W-W/100*100),E)+"CM":RETURN
1300 POKE&H4000,&HF:Z=15:LOCATE&H8,22:PRINT"ｼﾞｬﾝﾌﾟ ｶｸﾄﾞ"Z
1310 IFINKEY$=" "THENZ=Z+&H3:LOCATE16,22:PRINTZ:IFZ<75THEN1310
1320 H=Z:IFZ>45THENZ=90-Z
1330 L=P*&H3*Z/170:H=P*H/75:SPRITEA:SPRITEB:IFH<L THEN1350
1340 Q=-B:U=G:V=-H*G/L:S=(X+L)*G:X=X*G:Y=Y*G:I=&H4F:J=&H8:GOSUB1360:S=S+L*G:V=-V
:I=&H3F:J=&H9:GOSUB1360:GOTO1430
1350 Q=A:V=-G:U=L*G/H:S=(Y-H)*G:X=X*G:Y=Y*G:I=&H4F:J=&H8:GOSUB1360:S=S+H*G:V=-V:
I=&H3F:J=&H9:GOSUB1360:GOTO1430
1360 SPRITEE,X/G,Y/G:POKE&H4001,&HB5,I,J:X=X+U:Y=Y+V:IFX<2300THEN1380
1370 X=160:Y=Y+900:S=S+900:F=B:IFQ THENS=S-3040
1380 IFQ THEN1410
1390 IFABS(S-Y)>A THEN1360
1400 RETURN
1410 IFX<S THEN1360
1420 SPRITEE,X/G,Y/G:RETURN
1430 POKE&H4001,&H78,A,A:X=X/G:Y=Y/G:RETURN
```



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0									ル	イ	ー	ジ		ノ		3	ダ	ン	ト	ビ								
1									K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52								
2																												
3																												
4																												
5																												
6																												
7	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	M52	M52	G42	G42	G42	G42	G42	G42	G42	G42	G42
8	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
9																												
10																												
11																												
12										T	R	Y		N	O													
13																												
14																												
15																												
16																												
17																												
18	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	M52	M52	M52	M52	M52	M52	M52	M52	M52	M52	M52	M52	M52
19	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
20																												

きる方法はないかなと考えてみることだ。たいていは代わりの方法が見つかるものだ。だから、いいキャラクターなんてできないよーという前に、本当にできないのかどうか確かめてみよう。

⑶背景はどうなの

さて、キャラクターの重要性がわかったところで、次は背景を考えよう。背景は、キャラクターの活躍する舞台だ。だから、キャラクターに美人を用意するなら、やっぱり美人にふさわしい舞台を用意してやらないといけない。そしてこれは、画面美人化問題のもう1つの答えなんだ。ただ、本物の舞台設計がそう簡単ではないように、こちらの舞台も作るのは大変だ。それも美人にふさわしいようにだからね。僕が思うに、舞台設計に一番必要なのはセンスじゃないかな。センスのある人のつくったものは、やはりどこかが違うものね。そーか、センスか、それなら僕には無理だな、なんて思ったキミ、そんなことはない。ある程度のセンスなら誰でも身につけられるっていうのは、ファッションにしろデザインにしろいわれてること。もちろん背景作りでも同じだ。現にセンスのセの字もない僕でも背景らしきものぐらいは作れるもんね。

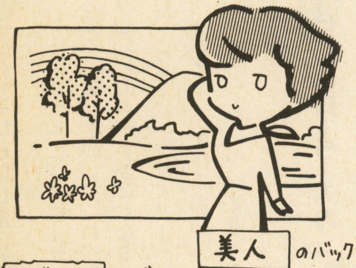
それじゃあ、どうすればセンスが身につくか。やっぱり、作ってみることだ。とにかく作る、このひとことだと思う。ただ、前にも言ったように、背景とはキャラクターの活躍する場であり、キャラクターを引き立たせるものだ。このごくごく基本的な事を忘れてはいけない。これがポイントだ。十分注意しよう。

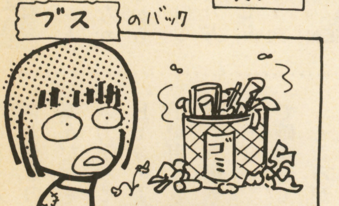
まあ、いろいろ試してみることだね。センスがあれば、すぐにいいものができるというだけで、センスがなくたって手間をかけてじっくり作ってやれば、いいものはできる。じっくり気長に取り組んでやろう。そうすれば、そのうちにきっとセンスは身につくハズだ。

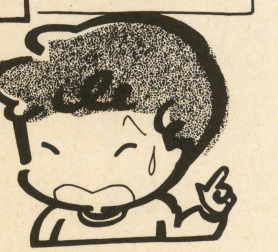
⑷遊び方

カーソルキーの→と、スペースバーを使う。ファンファーレがなったら、→キーを叩く。すると、ルイージが走り出す。ふみきり板のところまできたら、スペースバーを押そう。ジャンプ角度が表示され、スペースバーを押している間じゅう大きくなっていく。ここだと思った瞬間、スペースバーから手を離すと、そのとき表示された角度で、ルイージがジャンプ！

これをホップ・ステップ・ジャンプと3回行う。跳び終わると、跳躍距離が測定され、1トライが終了する。1ゲームで3回トライできる。途中でゲームをやめたいときは、ESCキーを押す。なお、飛距離は45度で跳んだときが一番出るようになっている。みんなで世界記録に挑戦してみるのも面白いぞ。



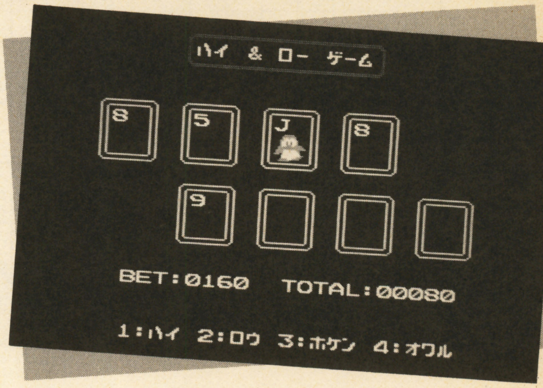




第3章
入力ルーチンここが肝心！

入力ルーチン作成のポイント大紹介

ゲームをやるのにワクワク気分の君たちも、ゲーム作りに闘志をもやす君たちも、最初は入力。これをさけては通れません。
ファミリーベーシックの場合においても、そこのところは同じ。ここの章では入力の方法とポイントとなる部分。そしてその後におこなわれるチェックの大切さについてふれてみよう。



⑴入力しないとはじまらない

ゲームのなかで入力ルーチンも必要不可欠なものだ。これがなかったらゲームができなくなるからね。ここでは、ファミリーベーシックでゲームを作る際の入力方法とポイントについて考えてみよう。

⑵コントローラー入力

ゲームの種類、入力装置の違いで入力方法は異なってくる。たとえばシューティングゲーム。この手のゲームでは、入力装置にコントローラーを用いるのがほとんどだ。ベーシックのコマンドでいえば、STRIGやSTICKといった関数を用いることになる。今、✚ボタンで左右に動き、Aボタンでレーザーを発射するゲームを作ってみる。このプログラムの入力ルーチンはどうなるだろうか。

図1を見てみよう。単純に考えればこうなるんじゃないかな。もちろん、このままでは無限ループになっちゃうし、キャラクターの動きの問題から他にいろいろな処理が付け加わるとは思うけど、基本はこれだと思う。

⑶キーボード入力

シューティングゲームの例をみたけど、入力装置としてはコントローラーの他にもキーボードがあるんだ。そしてキーボードからの入力をみる命令としては、INKEY$・INPUT・LINPUTが用意されている。各命令がどういう働きをするかはここでは説明しない。マニュアルを見て自分で確認してほしい。

ところが実際のゲームでは前の2つばかりが使われ、ほとんどLINPUTは使われない。よほどのことがないかぎり、INPUTでまにあってしまうからなんだろう。まあ使えないことはないから使ってみるといいんじゃないかな。少し工夫がいりそうだけどね。

さて前の2つの命令だが、どのように使われているだろうか。実例として、「ハイ&ロー」のプログラムリストを見てみよう。40行と2020行付近が入力ルーチンで、INKEY$が使われているね。この命令のいいところは、何か入力する際にいちいちリターンキーを押さなくてすむことだ。この利点は大きい。入力のたびに、リターンキーを押すのは、かなり面倒で、ゲームの流れを止めがちだ。だからこれをしなくてすむのは大いに助かる。

しかし、欠点もある。1文字しか読めないし、入力文字を表示しない。だから何文字も入力しなければならないときは、そのままでは使えない。もちろんプログラムを作ってやればいいんだけど面倒だ。

そこで登場するのがINPUTだ。あいにくと「ハイ&ロー」では2ヵ所の入力ルーチンともINKEY$を作ってしまっているけど、2020行のほうはINPUTを使うこともできる。実際にやってみるといいだろう。INKEY$とINPUTの違いがよくわかるし、やっぱりゲームではINKEY$のほうがいいって思うから。

⑷チェックの重要性

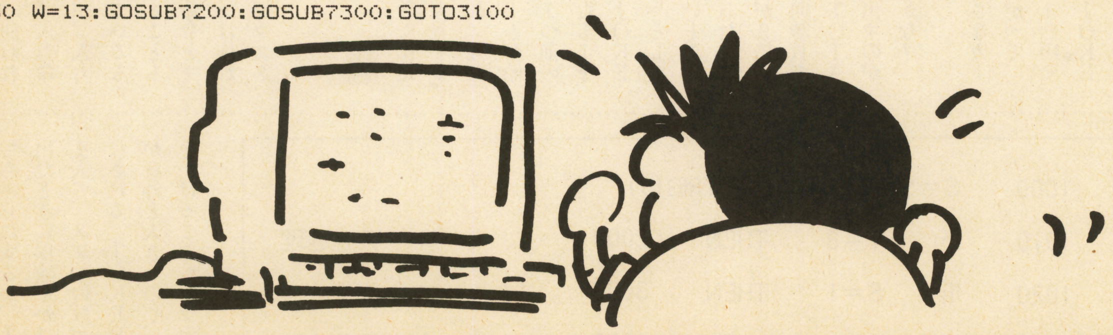
入力命令についていろいろみてきたが、もう1つ重要なことが残っている。それは入力チェックだ。これを忘れてはいけない。

人間は、すぐに間違いを犯すんだよね。当然入力も間違える。たとえば入れるべきデータの数を間違えたり、数字を入れるところに文字を入れてしまったり…。こういう場合はベーシックのほうでチェックして、違っているのを教えてくれるからいいんだけど、そうでない場合は大変だ。

よくあるのが入力値に範囲があるのにそのチェックを忘れ、範囲

一発勝負！ハイ&ロー

```
30 CLS:LOCATE5,12:PRINT"HIGH & LOW GAME":LOCATE12,23:PRINT"HIT SPACE BAR";
40 W=RND(13):W$=INKEY$:IF W$<>" " THEN40
80 DIMCD$(4),CC(4)
90 FORI=0TO4:READCC(I),W:FORJ=0TO3:CD$(I)=CD$(I)+CHR$(W+J):NEXT:NEXT
95 DATA0,104,2,28,2,0,1,88,3,112
100 MO=100:BM=10:SPRITEON
110 PALETS 0,15,48,38,18:PALETS 1,15,48,21,18:PALETS 2,15,53,37,23:PALETS 3,15,4
8,39,22
1000 VIEW:GOSUB7500:GOSUB7100
1010 GOSUB7000:LOCATE5,22:INPUT"INSERT COIN ";BT:IF BT<0 OR BT>MO OR BT>BM THENB
EEP:GOTO1010
1020 IF BT=0 THEN6010
1030 MO=MO-BT:GOSUB7100:GOSUB7000:W=13:GOSUB7200:WN=0:GOSUB7300:NO=NN
2010 WN=WN+1:W=14:GOSUB7200:LOCATE4,22:PRINT"1:ﾊｲ 2:ﾛｳ 3:ﾎｹﾝ 4:ｵﾜﾙ ";
2020 W$=INKEY$:IF W$="" THEN2020
2025 W=VAL(W$):IF W<1 OR W>4 THENBEEP:GOTO2020
2030 IF W=4 THEN5500
2040 HL=W:GOSUB7300
3005 FC=1:IF NN=14 THEN8010
3010 IF NO>NN THEN FC=2
3020 IF FC<>HL THEN5010
3100 BT=BT+BT:GOSUB7100:IF BT+MO>30001 THEN6100
3110 NO=NN:IF WN=7 THEN5500
4010 GOTO2010
5010 GOSUB7000:LOCATE9,22:PRINT"ｻﾞﾝﾈﾝ ﾃﾞｼﾀ":PLAY"T101C5DCDC":PAUSE100:BT=0:IF MO
=0 THEN6010
5020 GOTO1000
5500 GOSUB7000:LOCATE5,22:PRINT"ｵﾒﾃﾞﾄｳ! ｱﾅﾀ ﾉ ｶﾁﾃﾞｽ":PLAY"T103C5ECEC":MO=MO+BT:B
T=0:GOSUB7100:PAUSE60
5520 GOTO1000
6010 GOSUB7500:CLS:LOCATE5,12:PRINT"*** GAME OVER ***":END
6100 GOSUB7500:CLS:LOCATE5,12:PRINT"*** ｳ ﾁ ﾄﾞ ﾒ ***":END
7000 LOCATE2,22:PRINT"                            ":RETURN
7100 W=BT:X=8:Y=18:F=4:GOSUB7110:W=MO:X=20:F=5:GOSUB7110:RETURN
7110 W$=STR$(W):W$=RIGHT$(W$,LEN(W$)-1):W$=RIGHT$("0000"+W$,F):LOCATEX,Y:PRINTW$
:RETURN
7200 NN=RND(W)+1:IF NO=NN THEN7200
7210 RETURN
7300 RESTORE7400:FOR I=1 TO NN:READ D$:NEXT:X=3+WN*5:Y=6:IF WN>3 THEN X=X-15:Y=1
2
7310 LOCATEX,Y:PRINT"   ":LOCATEX,Y+1:PRINT"   ":LOCATEX,Y+2:PRINT"   ":LOCATEX,Y:P
RINTD$:PLAY"T104C1"
7320 W=NN:IF NN=1 THEN W=15
7330 IF W<11 THEN RETURN
7340 W=W-11:DEFSPRITE WN,(CC(W),1,1,0,0)=CD$(W):SPRITE WN,X*8+16,Y*8+32:RETURN
7400 DATA A,2,3,4,5,6,7,8,9,10,J,Q,K,JK
7500 FORI=0TO7:SPRITEI:NEXT:RETURN
8010 IF HL<>3 THEN5010
8020 GOSUB7000:LOCATE5,22:PRINT"ｽﾊﾞﾗｼｲ ｶﾝ ｦ ｼﾃﾏｽﾈ !?":PLAY"T102M1Y1C9E9E7DC9EDE7
DC9":BT=BT*4:WN=WN+1:IF WN=8 THEN5500
8030 W=13:GOSUB7200:GOSUB7300:GOTO3100
```



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0								K72	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L02								
1								K62	ハ	イ		&		ロ	ー		ゲ	ー	ム	K62								
2								L12	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L22								
3																												
4																												
5			I60	J30	J30	I70		I60	J30	J30	I70		I60	J30	J30	I70		I60	J30	J30	I70							
6			J20	L30	L70	J20		J20	L30	L70	J20		J20	L30	L70	J20		J20	L30	L70	J20							
7			J20	L40	L40	J20		J20	L40	L30	J20		J20	L40	L40	J20		J20	L40	L40	J20							
8			J20	L50	M00	J20		J20	L50	M00	J20		J20	L50	M00	J20		J20	L50	M00	J20							
9			J00	J30	J30	J10		J00	J30	J30	J10		J00	J30	J30	J10		J00	J30	J30	J10							
10																												
11								I60	J30	J30	I70		I60	J30	J30	I70		I60	J30	J30	I70		I60	J30	J30	I70		
12								J20	L30	L70	J20		J20	L30	L70	J20		J20	L30	L70	J20		J20	L30	L70	J20		
13								J20	L40	L40	J20		J20	L40	L40	J20		J20	L40	L40	J20		J20	L40	L40	J20		
14								J20	L50	M00	J20		J20	L50	M00	J20		J20	L50	M00	J20		J20	L50	M00	J20		
15								J00	J30	J30	J10		J00	J30	J30	J10		J00	J30	J30	J10		J00	J30	J30	J10		
16																												
17																												
18					B	E	T	:							T	O	T	A	L	:								
19																												
20																												

外の値を取りこんじゃうというヤツ。入力データが計算に使われるのなら、メチャクチャな結果が出るだけでそれほど問題はないが、その入力値で次の処理を決めるなんて場合はとんでもないことになる。

ベーシックではそれほどひどいことにはならないが、マシン語プログラムだと、暴走してプログラム自身を壊したり、最悪の場合にはマシンそのものを壊したりしちゃうことがあるんだよね（もちろん大爆発をするようなことはないよ）。だから入力チェックは絶対必要だ。

「ハイ&ロー」でももちろん入力チェックをしている。40行の入力は何が入力されてもいいから別にチェックをしてないが、2020行のほうは範囲のある入力を要求しているから、チェックをしている。2025行がそれだ。INKEY$は文字としてデータを取りこむから、1度それを数字に変換する。そして次にここでの入力は1から4まででなければならないから、入力データがその範囲にあるかどうかをチェックしているんだ。そしてチェックに引っ掛かった場合は、ブザーをならして注意を促し、再入力させている。こうして誤動作を防いでいるわけだ。ちなみに2025行のIF文をとってしまうと、最後の判定（3020行）でとんでもない判定をする可能性がでてくる。実際にどうなるかやってみるといいかもしれない。

⑸入力もチェックも大切なんだ

ファミリーベーシックでの入力について考えてきたが、ここでまとめておこう。ファミコンの場合、入力装置はコントローラーとキーボードの2つだ。それぞれに対して入力用の命令が用意されていて、装置・用途によって使い分けてやらなければならない。

キーボードからの入力をうける命令は3種類だが、よく使われるのはそのうちの2つだ。入力命令をよく知るのも大切だが、入力データがこちらの意図しているものかどうかをチェックすることも重要だ。基本的には入力命令をマスターすれば入力ルーチンは作れるはずなので、がんばってマスターしよう。

⑹遊び方

INSERT COINと出たら、賭けるメダルの数を入力する。するとカードが1枚開かれ、入力待ちとなり、ここからゲームが始まる。次のカードがその1つ前のカードより大きいと思ったら1を、小さいと思ったら2を、JOKERが出ると思ったら3を、そしてこれ以上の勝負は避けたいと思ったら4を入力する。1回あたるごとに、賭けたメダルの数が2倍になり、1回の賭けで、7回勝負できる。これを繰り返してTOTALを増やし、打ち止めを出すのが目的。でもその前に破産かな？

☆カードについて☆

Aは1とされ、一番小さなカードとして扱われる。したがって、カードの順位はA・2・3…10・J・Q・Kの順となる。JOKERは坊主めくりの坊主と同様に、出たとたんに今までの勝負をすべてちゃらにするが、JOKERが出たときに保険をかけておけば大丈夫で、メダルの数が一挙に8倍にされる。JOKERの出る確率が意外と大きいので、うまく保険をかけることがコツのようだ。

```
1000  S=STICK(0) : T=STRIG(0)
1010  IF   T=8   THEN   2000   :'  レーザー処理
1020  IF   S=1   THEN   3000   :'  右へ移動
1030  IF   S=2   THEN   4000   :'  左へ移動
1040  GOTO  1000
```

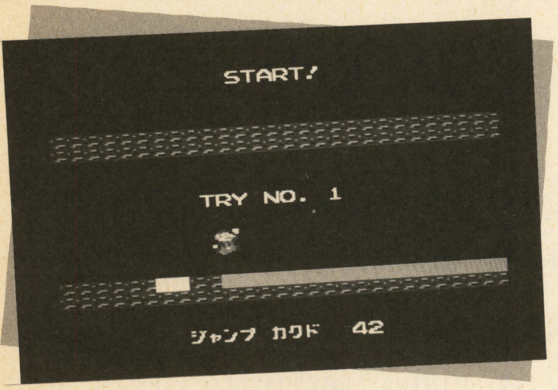
図1プログラム例

中級編

第4章

スプライトがおもしろい！

画面を楽しくみせる秘密兵器がこれだ



さあ、いよいよ面白ゲーム作りの主役登場。おまたせしましたスプライトです。

スプライトを使いこなせば画面は君の思いのまま。走り回る！飛び回る画面の中のキャラクター。やっぱりゲームはこうでなければウソだ。

さあ、スプライト機能で元気なゲームをつくってみよう。

⑴さあスプライト登場だ

ゲームプログラムの主役といえば、そう、スプライトだ。スプライトを使うとキャラクターを動かすのがかなり楽になるが、普通のパソコンにはこのスプライト機能はついていないんだよね。だからパソコンでキャラクターを動かすというのはけっこうたいへんだ。消しては描き、消しては描きを繰り返して、キャラクターが動いているように見せる。だから、キャラクターが重なったりすると、とたんに処理が複雑になっちゃうんだよね。

でも、スプライト機能があればそんな心配はまったくいらない。優先順位を決めてやれば、自動的に重ね処理をやってくれるし、移動も次の位置の指定だけでいい。スプライト機能はゲームプログラマーの秘密兵器といったところだ。

ここでは、このスプライトの使い方を勉強してみることにしよう。

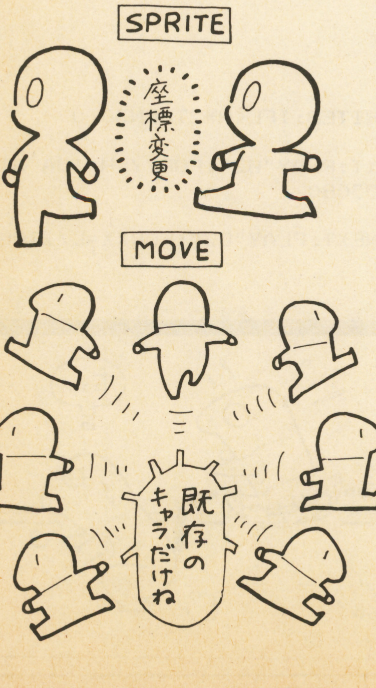
⑵使い分けるには

まずスプライトについての基礎知識だ。ファミリーベーシックではスプライトを扱うのに、2種類の命令が用意されている。そう、SPRITEグループとMOVEグループだ。各グループ内で16×16のスプライトが8枚使え、両グループは同時に使えるから計16枚のスプライトが1画面上で使える。各グループにはかなり特徴があり、慣れが必要だ。このグループの使い分けができるようにならなければ、スプライトに関して1人前とはいえない。

ここで両グループの特徴を挙げてみよう。SPRITEグループは、8×8単位、または16×16単位でキャラクターの設定ができ、キャラクターの移動はプログラムで表示座標(ざひょう)を少しずつ変えてやることで実現できる。そのため、かなり細かい動きができるが、処理スピードが遅いとギクシャクした動きになってしまう。

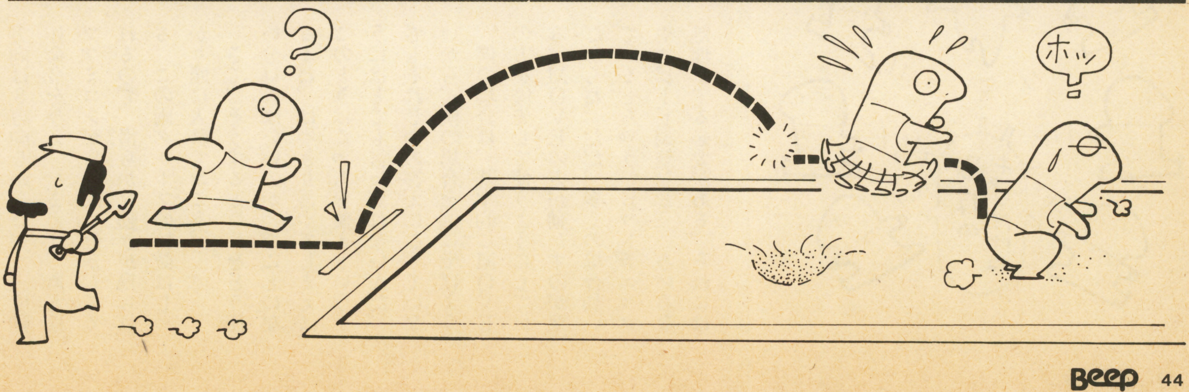
一方、MOVEグループは任意のキャラクター作成はできないが、ROM中の既存キャラクターを指定して動かすことができる。動作方向は8方向しかないが、一度動作命令を与えてしまえば、あとはファミコンが自動的に動かしてくれるので、滑(なめ)らかな動きが得られる。この両グループの特徴はしっかり頭に入れておこう。

さて、そうするとこれをどう使い分ければいいのかってことになるよね。はっきり言ってしまえば、このときはこっちを使えというようにはとても言い切れないんだ。でもそれじゃあ答えになっていないから、僕の場合を例に挙げよう。僕の場合は、基本的にMOVEの方を優先して使うようにしている。MOVEの方が動かすのが楽だからね。ただ、多くの場合動いているとちゅうでヒットチェックやスピード調整など、いろいろな処理をする必要がある。それには



マリオの走り幅(はば)とび

```
100 A=0:B=1:DIMR(2):CGSETB,B:PALETS A,15,54,22,2:PALETS B,15,48,38,18:PALETB A,1
5,48,41,2
200 DEFSPRITEA,(A,B,A,B,A)=CHR$(B)+CHR$(A)+CHR$(3)+CHR$(2):DEFSPRITEB,(A,B,A,B,A
)=CHR$(5)+CHR$(4)+CHR$(7)+CHR$(6):DEFSPRITE2,(A,B,A,B,A)=CHR$(13)+CHR$(12)+CHR$(
15)+CHR$(14):SPRITEON
900 T=A
1000 VIEW:LOCATE9,12:PRINT"TRY NO.";T+B
1020 X=16:Y=64:D=A:P=B:C=A:J=3:F=A:K=A
1030 SPRITEA,X,Y
1040 LOCATE11,3:PRINT"START!"
1050 IFINKEY$=""THEN1050
1110 C=C+B:IFC>10ANDP>A THENP=P-B:C=A
1120 W=ASC(INKEY$):IFW=32ANDF=B THEN1500
1130 IFW<>A ANDW<>28THEN1110
1140 IFW=K THEN1110
1150 K=W:J=J+B:IFJ>20THENJ=3:P=P+B
1160 X=X+P:IFF=B ANDX>75THEN1600
1170 IFX>230THEN1200
1180 SPRITED:D=J MOD2:SPRITED,X,Y:C=A:GOTO1110
1200 F=B:X=16:Y=150:GOTO1180
1500 GOSUB3000:R=(X-68)*25/4:GOSUB2900:I=RND(B):J=RND((P*3/5+B)*3):IFI=A THENJ=-
J
1510 R=R+J:IFR<0THEN1600
1520 DEFMOVE(A)=SPRITE(4,3,2,(X-68)/2,A,B):POSITIONA,68,150:MOVEA:LOCATE5,12:PRI
NT"ﾀﾀﾞｲﾏ ﾉ ｷﾛｸ      "
1530 W=(XPOS(A)-70)*25/4:GOSUB2800:LOCATE16,12:PRINTW$:PLAY"O4C0E0":IFMOVE(A)=-B
 THEN1530
1580 W=R:GOSUB2800:LOCATE16,12:PRINTW$
1590 PAUSE150:GOTO2000
1600 GOSUB2900:R=A:LOCATE11,3:PRINT"FOUL!  ":PLAY"T101B5AGDEDC":PAUSE100
2000 POSITIONA,A,A:ERAA:SPRITEA:SPRITEB:SPRITE2:R(T)=R:IFMX<R THENMX=R
2010 T=T+B:IFT<3THEN1000
2500 CLS:LOCATE7,3:PRINT"TODAY'S BEST":W=MX:GOSUB2800:LOCATE9,5:PRINTW$:LOCATE7,
7:PRINT"YOUR RECORDS":FORY=A TO2:W=R(Y):GOSUB2800:LOCATE7,9+Y*2:PRINTY+B"ｶｲﾒ:"W$
:NEXT
2510 LOCATE7,22:PRINT"REPLAY? (Y/N)"
2520 W$=INKEY$:IFW$=""THEN2520
2530 IFW$="Y"THEN900
2540 CLS:LOCATE5,12:PRINT"*** GAME OVER ***":END
2800 W$=RIGHT$(STR$(W/100),2)+"M "+RIGHT$(STR$(W-W/100*100),2)+"CM":RETURN
2900 LOCATE11,3:PRINT"        ":RETURN
3000 I=5:PLAY"T104":LOCATE8,22:PRINT"ｼﾞｬﾝﾌﾟ ｶｸﾄﾞ "I
3010 IFINKEY$<>" "THEN3040
3020 I=I+B:LOCATE17,22:PRINTI:IFI=85THEN3040
3030 GOTO3010
3040 H=I:IFI>45THENI=90-I:H=H+4
3050 L=I/5*P*3+(I-I/5*5)*P*3/5:H=H/5*P*2/3:SPRITEA:SPRITEB:IFL<H*2THEN3070
3060 X=X*10:XX=A:IFH>A THENXX=L*5/H
3065 FORYY=B TOH:X=X+XX:SPRITE2,X/10,Y-YY:PLAY"F0":NEXT:PLAY"O2":FORYY=H TOA STE
P-B:X=X+XX:SPRITE2,X/10,Y-YY:PLAY"F0":NEXT:X=X/10:GOTO3080
3070 Y=Y*10:YY=A:IFL>A THENYY=20*H/L
3075 FORXX=B TOL/2:Y=Y-YY:SPRITE2,X+XX,Y/10:PLAY"F0":NEXT:PLAY"O2":FORXX=L/2TOL-
B:Y=Y+YY:SPRITE2,X+XX,Y/10:PLAY"F0":NEXT:X=X+L
3080 RETURN
```



SPRITEの方がずっと扱いやすいんだ。だから、必然的にSPRITEを使うことが多い。つまり僕の場合は、とちゅうで処理を要求しない場合はMOVEを、それ以外はSPRITEを使うようにしているんだ。ただこれはあくまでも僕の場合だから、まねをする必要はない。キミたちが、自分に合った基準を見つける際の参考にしてもらえばいいんだ。

③実際例

次にくる問題は、具体的にどう使うかということだ。ここでは「マリオの走り幅とび」と「スロットマシン」の2つを具体例として取り上げた。前者は普通の使い方をしている例で、後者はちょっと変わった使い方をしている例だ。それぞれのゲームでスプライトをどう使っているかよく見てほしい。

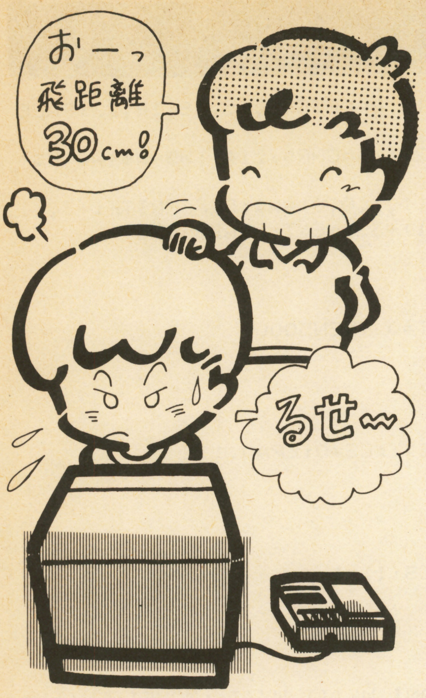
まず最初に普通の使い方の例。「マリオの走り幅とび」だ。このゲームでは、SPRITEグループでマリオを表示し、MOVEグループでペンペンを表示している。まず200行でマリオのキャラクターパターンを定義している。0番と1番が走り用で、2番がジャンプだ。このゲームではカーソルキー→をたたくとマリオが走るわけだが、これは、→キーの入力ごとに表示するスプライトを変えているだけなんだ。つまり、0番と1番を交互に表示しているわけ。もちろん片方を表示するときは忘れずにもう片方を消しているけどね。ただそれだけだが、けっこう走っているように見える。なかなかうまい使い方だ。ジャンプのときは、走り用の0番と1番のどちらも消して2番のジャンプ用スプライトを移動させている。

このようにマリオに関するものは全部SPRITEグループを使っているが、これはMOVEだと細かな操作（具体的にはキー入力をマリオのスプライトに変える処理が大変なのと、ジャンプ角度はMOVEを使うと表わせないということ）がやりにくいからなんだ。

一方、ペンペンのほうにはMOVEを使っている（1520行）。もちろんSPRITEでやるのも可能だ。けれどもそうすると、MOVEの場合よりもプログラムが増えちゃうし、メリットが全然ないので使っていない。結局スプライトという点からみるとこのゲームは、SPRITEグループで3枚、MOVEグループで1枚の計4枚しかスプライトを使っていない。シンプルな構成だけど、けっこう面白い。いたずらに複雑なゲームよりシンプルなゲームのほうが面白いといういい例だね。

それではちょっと変わった使い方をしている「スロットマシン」のほうをみてみよう。110行と130行でSPRITEグループのスプライト定義、2300行でMOVEグループのスプライト定義をしている。ゲームをやってみると、スロットマシンのドラムをMOVEで表示しているのはすぐにわかる。でもSPRITEが何をしているのかは、ちょっと見ただけではわからないんじゃないかな。エッ、わかってるって？ さすがBEEPERだ。そう、それはスロットマシンの窓の上と下で使われているんだね。

これはSPRITEグループのほうがMOVEグループのスプラ



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0																												
1																												
2																												
3																												
4																												
5																												
6																												
7	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
8	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
9																												
10																												
11																												
12																												
13																												
14																												
15																												
16																												
17																												
18	G42	G42	G42	G42	G42	G42	M50	M50	G42	G42	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
19	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
20																												

カジノもまっ青?スロットマシン

```
100 CGSET1,1:PALETS 0,15,48,38,18:PALETS 1,15,15,15,15:PALETS 2,15,48,18,22:PALE
TS 3,15,48,39,22
110 DIMWX(2),WY(2),WC(2),WT(2),SP(2),CN(2),CS(2),CL(2):W$=CHR$(214):W$=W$+W$+W$+
W$
130 FOR I=0 TO 2:READWX(I),WY(I),WT(I):DEFSPRITEI,(1,1,1,0,0)=W$:DEFSPRITEI+3,(1
,1,1,0,0)=W$:SPRITEI,WX(I),WY(I):SPRITEI+3,WX(I),WY(I)+32:NEXT:READBS,MO
140 VIEW:SPRITEON:GOSUB6100
300 DATA 88,31,5,120,31,7,152,31,9,4,50
1000 '
1010 GOSUB6000:LOCATE8,22:INPUT"INSERT COIN";BT:IF BT=0 THEN5000
1020 IF BT>50 THEN BEEP:GOTO1010
1030 IF BT>MO THEN BEEP:GOTO1010
1040 MO=MO-BT:GOSUB6100:W=1:GOSUB6200:GOSUB6000
2000 '
2010 FOR I=0 TO 2:SP(I)=1:WC(I)=0:CS(I)=0:CL(I)=16:NEXT:CS=0:ERA0,1,2
2100 IF CS=3 THEN3000
2110 FOR I=0 TO 2:IF MOVE(I)=-1 THEN 2400
2115 IF CS(I)=1 THEN2400
2120 WC(I)=WC(I)+1:IF WC(I)<WT(I) THEN2200
2130 SP(I)=SP(I)+1:IF SP(I)=BS THEN CS=CS+1:CL(I)=8:CS(I)=1
2200 W=RND(10):IF W<4 THEN W=1:CO=0:CN=4:GOTO2300
2210 IF W<7 THEN W=2:CO=3:CN=5:GOTO2300
2220 IF W=9 THEN W=4:CO=2:CN=14:GOTO2300
2230 W=3:CO=3:CN=13
2300 DEFMOVE(I)=SPRITE(CN,5,SP(I),CL(I),1,CO):POSITIONI,WX(I),WY(I):MOVEI:CN(I)=
W
2400 NEXT:GOTO2100
3000 IF MOVE(0)=-1 OR MOVE(1)=-1 OR MOVE(2)=-1 THEN3000
3010 IF NOT((CN(0)=CN(1)) OR (CN(1)=CN(2))) THEN MU=0:W=3:GOSUB6200:GOSUB7100:GO
TO3900
3020 MU=3:IF CN(0)=CN(2) THEN MU=MU+MU
3030 MU=MU*CN(1):W=2:GOSUB6200
3100 W=MU*BT:MO=MO+W:W$=STR$(W):W$=RIGHT$(W$,LEN(W$)-1):X=13:IF LEN(W$)=5THEN X=
12
3110 LOCATEX,18:PRINTW$
3200 GOSUB7000:GOSUB6100
3900 IF MO<1 THEN5000
3910 IF MO>30001 THEN5500
4000 GOTO1000
5000 SPRITEOFF:CLS:LOCATE5,12:PRINT"*** GAME OVER ***":END
5500 SPRITEOFF:CLS:LOCATE9,12:PRINT"ｺﾘｬｰﾏｲｯﾀ ...":END
6000 LOCATE7,22:PRINT"                              ":LOCATE12,18:PRINT"     ":RETURN
6100 LOCATE23,17:PRINT RIGHT$("    "+STR$(MO),5):RETURN
6200 X=11:ONW GOTO6210,6220,6230
6210 W$="       ":GOTO6240
6220 W$=STR$(MU):W$="X"+RIGHT$(W$,LEN(W$)-1):X=13:GOTO6240
6230 W$="SORRY!"
6240 LOCATE X,14:PRINTW$:RETURN
7000 PLAY"T1Y2M1V503GRDREGDA":RETURN
7100 PLAY"T1Y2M1V502#GAREDBC":RETURN
9000 'PROGRAM  VERSION 1.0
9010 'PROGRAMED BY HOOCHIN
```



イトよりも優先順位が高いという、マニュアルには出ていない（やればわかるんだけど）機能を使ったものなんだ。上下を隠してやることで、キャラクターが少しずつ出たり入ったりすることになる。それで、ドラムが回転しているように見えるんだ。

それぞれの命令の使い方は、ごくシンプルな使い方しかしていないがうまく組み合わせてすばらしい動作を可能にしている。ちょっとした工夫によってこんなにも面白い効果がだせるんだというい例だ。キミたちも、ぜひこのようなちょっとした工夫をしてみて、ゲームを面白くしてみよう。

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0								K72	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L02							
1								K62													K62							
2								K62	K72	K52	K52	L02	K72	K52	K52	L02	K72	K52	K52	L02	K62	K72	K52	K52	L02			
3								K62	K62			K62	K62			K62	K62			K62	K62	L12	L02	K72	L22			
4								K62	K62			K62	K62			K62	K62			K62	K62		K62	K62				
5								K62	L12	K52	K52	L22	L12	K52	K52	L22	L12	K52	K52	L22	K62		K62	K62				
6								K62													K62		K62	K62				
7							K72	K12	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K12	L02	K62	K62				
8							K62															K62	K62	K62				
9							K62				N	I	N	T	E	N	D	O				K42	L22	K62				
10							K62															K42	K52	K22				
11							L12	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L22						
12										K62									K62									
13										K62									K62									
14										K62									K62				Y	O	U	R		
15										K62									K62					C	O	I	N	
16										K62		K72	K52	K52	K52	K52	L02		K62									
17										K62		K62					K62		K62									
18										K62		K62					K62		K62									
19							K72	K52	K52	K12	K52	K12	K52	K52	K52	K52	K12	K52	K12	K52	K52	L02					マ	イ
20							L12	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	L22						

⑷スプライトの道は険し

結局、スプライトをうまく使うには、2つのグループの特徴を知って、うまく組み合わせたり使い分けたりすることが必要だ。命令がわかったというだけではダメだ。これは初心者にとっては大きな壁だけど、他人のプログラムをみたり、自分で作ったりすることで越えられる。がんばってほしい。

⑸遊び方（マリオの走り幅とび）

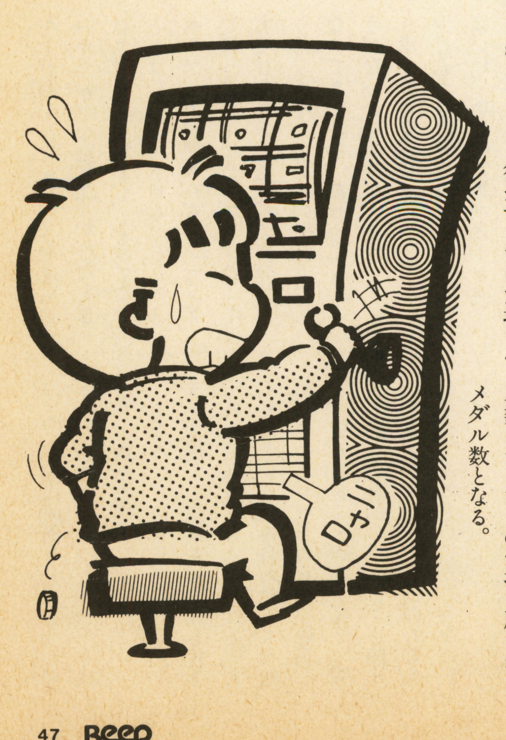
キーボードのカーソルキー→とスペースバーを使い、STARTの表示が出たら、ひたすら→キーをたたいてマリオを走らせよう。ふみきり板のところでスペースバーを押していると、ジャンプ角度が表示される。角度は、スペースバーを押してやると増えるので、ここだと思った瞬間に手を離せば、そのとき表示されていた角度で、マリオがジャンプ！着地するとペンペンが登場して、飛距離を測定する。これで1トライが終了し、全部で3回トライできる。さらに、この3回の記録は最後にまとめて表示され、最高記録はTODAYS'BESTに登録される。ゲーム自体は1人用だが、何人かで飛距離を競いあうというような遊び方もできる。

⑹遊び方（スロットマシン）

遊び方は実に簡単。まずINSERT COINと出るので、賭けたいメダルの数を入力してやる。するとスロットマシンが動き始め、しばらくすると自動的にとまり、役ができているかどうか判定してくれる。うまく役ができていた場合には、その役に応じて払い戻しがある。これで一回のプレイは終了。これを何度も繰り返して、自分のメダルの数を増してやればいい。もしもメダルの数が不幸にも0になってしまったら、その時点でGAME OVERになってしまう。またメダルが3万枚を超えた場合も打ち止めでGAME OVERだ。メダルは一度に最高50枚まで賭けられ、途中でゲームをやめたいときには賭けるメダル数を0にしてやる。

☆役☆

役は2種類。3つともそろったときが3倍、2つのときが2倍だ。これにキャラクターの出現頻度に応じた倍率が賭けられ、最終倍率となる。そしてこれに賭けたメダル数をかけたものが払い戻されるメダル数となる。

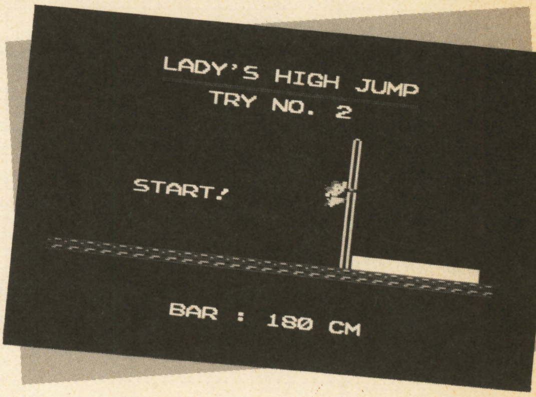


中級編

第5章
キャラクター操作術

スプライトで自由自在に画面を演出

前章で学んだスプライト。実際例を見ながら、確実に身につけてしまおう。ここの章で出てくるゲーム〝レディの走り高とび〟は、スプライト機能を効果的に使いこなしたおもしろゲーム。息づまる走り高とびのステップの瞬間を熱く再現している。さあ、この楽しさをじっくり味わってください。



(1)キャラクターを動かしてみるか

前章ではスプライトの利用について考えてみた。そこで、本章ではスプライトのキャラクターを実際に動かすにはどうすればいいのかを考えてみよう。例として「レディの走り高とび」を取り上げ、このレディの走りの部分を徹底的にみることにしよう。

まだこのゲームをやったことがない人は、本文を読む前にゲームを一度やってほしい。ではさっそく説明に入ろう。

(2)実際はこうだ

キャラクターの設定は、１０１０行で行っている。ＰＯＫＥ文で直接ワークエリアにキャラクターデータをセットしているので、少々わかりにくいが、アドレスの次からがデータだ。０番から４番まではレディに使い、０番と１番が走り用、２番がジャンプ用、３番と４番がバーに当たったとき用だ。５番はバーそのもの。これらを場合に応じて動かしてやるわけだ。

注目のレディの走りを担当するＲＵＮルーチンは、１０７０行から１１３０行にかけてのプログラムだ。この部分のフローチャートは図１のようになる。これで→キーが叩かれたらレディを走らせ、→キーが一生懸命叩かれたときはスピードをＵＰし、そうでないときはＤＯＷＮさせる。そしてスペースバーが押されるとジャンプ処理へ飛ぶ。これをプログラムで実現したものが先ほどの部分だ。一体どうなっているのか知りたくなってきたかな。あせってもしょうがないから気楽にいこう。

走っている感じは、パターンが少し違うものを連続して表示すればいい。いわゆるアニメーション処理だ。まあこれは、どんなＴＶゲームでもやっていることだね。

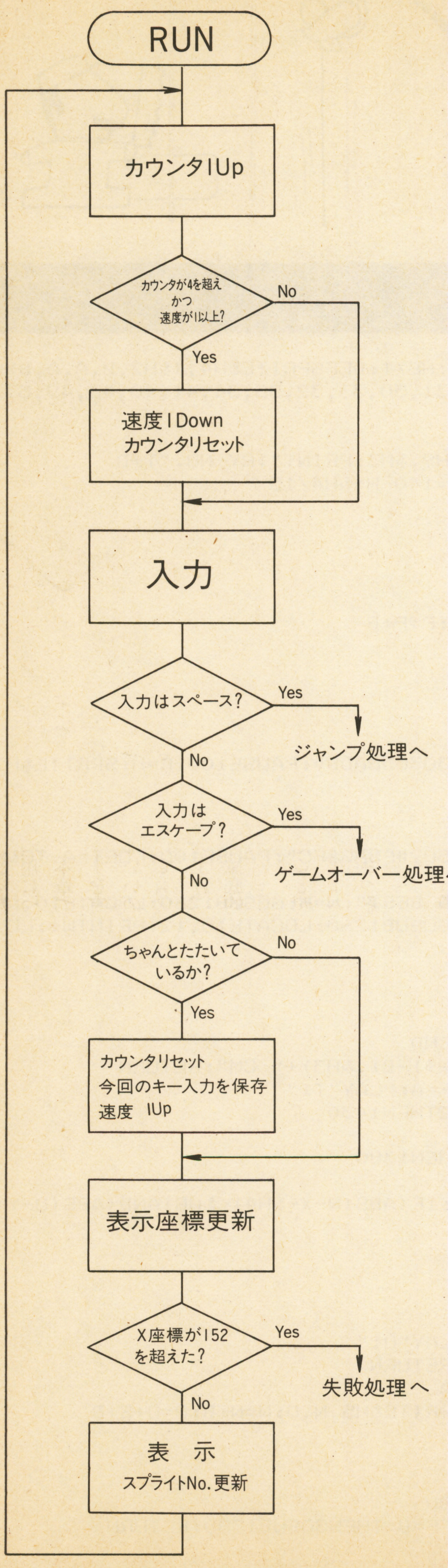
ファミリーベーシックではＷＡＬＫとしてレディに３つのパターンが用意されている。本ゲームではこのうち、ＷＡＬＫ１とＷＡＬＫ２の２パターンを使っている。この２パターンを交互に見せながら、走っているように見せるわけだ。それが１１２０行と１１３０行だ。１１２０行で前回表示したスプライトを消し、表示スプライトナンバーを更新する。そして１１３０行で新しいスプライトを表示している。２パターンの切りかえで走りを表現しているので、実際に画面を見ると少しギコチなさは残っているけど、それほど不自然な感じはしない。

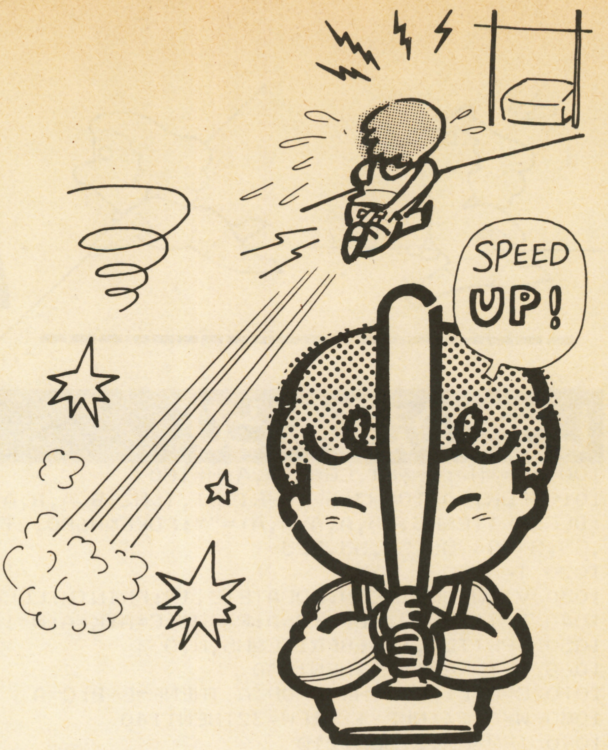
ちなみにＰＯＫＥ文のパラメータを変更してやると、ＷＡＬＫ１とＷＡＬＫ３、ＷＡＬＫ２とＷＡＬＫ３という組み合わせ方も可能だ。またプログラムを少し修正して、ＷＡＬＫ１・ＷＡＬＫ２・ＷＡＬＫ３の全部のパターンを使うこともできる。腕に自信のある人はやってみるといいんじゃないかな。

さて走りの基本的部分はわかった。次はスピード調整だ。スピードというパラメータ（プログラムではＰという変数で表す）をうまく画面に反映させるため、このプログラムではレディの次の位置を計算するのにこのスピード項を足



図1 RUNルーチンフロー



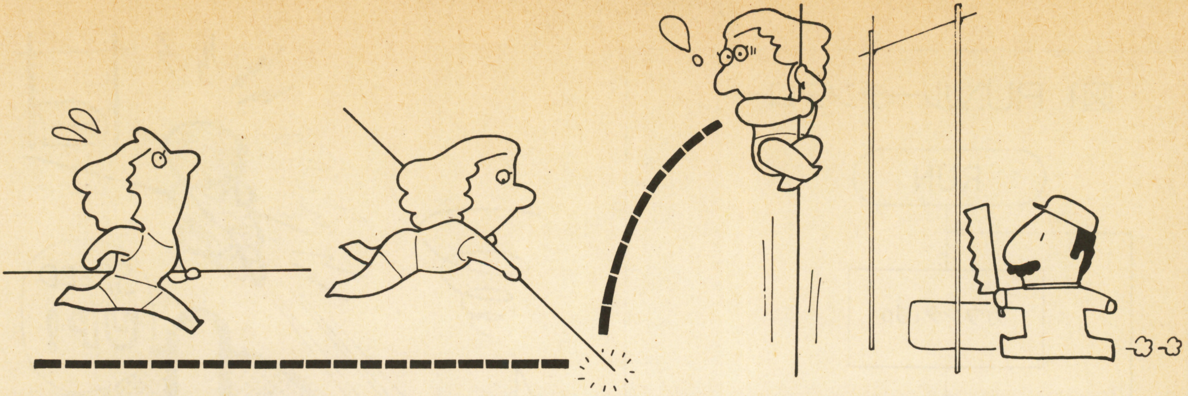


すようにしている。キー入力が正しく行われると、スピードを1つずつあげるようにして、位置計算にはこのスピードを10で割った値を使用している。これにより、10回すばやくキーを叩けばスピードが1だけあがるように見えるわけだ。これは1100行、1110行でプログラム化されている。

ここで、優秀な読者諸君は、こう言うだろう。確かにこれなら、→キーを叩くスピードがあがるが→キーを叩かないとスピードが落ちるはずなのに、これじゃあ落ちないよって。そう、確かにこのままではそのとおりになってしまう。しかし心配はいらない。1070行でちゃんとやっている。

ここでカウンタCを導入しているのに注目しよう。これは、このRUNルーチンのループ回数をカウントし、一定回数だけループした場合に、スピードを落とすという処理をさせるために導入されたものだ。このカウンタの導入で、きちんと入力されたときにこのカウンタをクリアしてやれば、正しい入力（連続して→キーを叩く）が行われているかぎりスピードは落ちないし、のんびり叩いたり、叩かなかったりした場合にはスピードが落ちるという処理が可能になるわけだ。本ゲームではこのループ回数を4回に設定しているが、これは変更可能だ。適当に変えてみて、このカウンタがゲームにどういう影響を及ぼしているかみてみるのも面白いよ。

以上が走りの部分の説明だ。わかったかな。そんなに難しくはないから、じっくりとプログラムをみて研究してほしい。このゲームの最大の特徴は、ジャンプルーチンにある。「マリオの走り幅とび」では、ごく単純なジャンプルーチンだったが、これはじつに複雑になっている。なにしろジャンプしながらキー入力を見て、それによって処理を変えなければならないし、バーに当たったかどうかもチェックしなければならない。プロ



レディの走り高とび

```
1000 A=&H0:B=&H1:CGSETB,A:H=180
1010 FORI=A TO&H3:DEFSPRITEI,(&H2,B,A,B,A)="":NEXT:DEFSPRITE&H4,(&H2,B,A,A,B)=""
:DEFSPRITE&H5,(A,A,A,A,A)="":POKE&H680,29,28,31,30,33,32,35,34,41,40,43,42,53,52
,55,54,54,55,52,53,223
1020 T=A
1030 VIEW:SPRITEON:LOCATE13,19:PRINTH:LOCATE&H9,&H3:PRINT"TRY NO."T+B
1040 X=16:Y=127:G=A:D=A:P=&H6:C=A:K=A:N=160:O=140-H*&H4/15
1050 SPRITEA,X,Y:SPRITE&H5,N,O
1060 IFINKEY$=""THEN1060
1070 C=C+B:IFC>&H4 ANDP>A THENP=P-B:C=A
1080 W=ASC(INKEY$):IFW=32THEN1140
1090 IFW=&H1BTHEN1210
1100 IF(W=28ANDK=A)OR(W=A ANDK=28)THENC=A:K=W:P=P+B
1110 X=X+P*&H3/30:IFX>152THEN1200
1120 SPRITED:D=D+B:IFD=&H2THEND=A
1130 SPRITED,X,Y:GOTO1070
1140 F=A:GOSUB1270:IFF=B THEN1200
1150 IFX<160 ORG=A THEN1200
1160 LOCATE&H5,10:PRINT"CLEAR !":PLAY"T1Y2M1V3O3E5GADB":PAUSE100:R=H:SPRITE&H2
1170 IFH>M THENM=H
1180 H=H+&H3:IFH<200THENH=H+&H2
1190 GOTO1020
1200 LOCATE&H5,10:PRINT" FAIL !":PLAY"T1Y3M1V5O2#F5AG#FC":PAUSE100:FORI=A TO&H5:
SPRITEI:NEXT:T=T+B:IFT<&H3THEN1030
1210 SPRITEOFF:CLS:LOCATE&H7,&H3:PRINT"TODAY'S BEST":W=M:GOSUB1260:LOCATE10,&H6:
PRINTW$:LOCATE&H7,&H9:PRINT"YOUR RECORD":W=R:GOSUB1260:LOCATE10,12:PRINTW$:LOCAT
E&H7,22:PRINT"REPLAY? (Y/N)"
1220 W$=INKEY$:IFW$=""THEN1220
1230 IFW$="Y"THENH=180:GOTO1020
1240 IFW$<>"N"THEN1220
1250 CLS:LOCATE&H5,12:PRINT"***GAME OVER***":END
1260 W$=STR$(W):W$=RIGHT$("00"+RIGHT$(W$,LEN(W$)-B),&H3)+" CM":RETURN
1270 V=P*&H3/&H2:U=V/&H4:SPRITEA:SPRITEB:D=B:F=A:PLAY"T1"
1280 I=A:S=-&H4:Z=B:PLAY"O3":GOSUB1320:IFF=&H2THEN1430
1290 IFF=B THEN1370
1300 I=A:S=&H4:Z=A:PLAY"O2":GOSUB1320:IFF=B THEN1380
1310 RETURN
1320 PLAY"F0":SPRITE&H2,X,Y:X=X+B:Y=Y+S:I=I+B:IF(ABS(N-X-&H4)<&H8)AND(ABS(O-Y+&H
4)<&H8)THENF=B:RETURN
1330 IFINKEY$=CHR$(28)ANDZ=B THENF=&H2:RETURN
1340 IFY<0 THENG=B
1350 IFI<U THEN1320
1360 RETURN
1370 SPRITE&H2:PLAY"O5":GOSUB1390:I=A:S=&H4:PLAY"O1"
1380 SPRITE&H2:SPRITE&H3:SPRITE&H4:GOSUB1390:RETURN
1390 PLAY"F0":SPRITED+&H3:D=D+B:IFD=&H2THEND=A
1400 SPRITED+&H3,X,Y:X=X+B:Y=Y+S:IFO<138THENSPRITE&H5,N,O:N=N+B:O=O+S+B
1410 I=I+B:IFI<U THEN1390
1420 RETURN
1430 V=(Y-127+P*&H2)*&H2/&H3:U=V/&H2
1440 I=A:S=-B:Z=A:GOSUB1320:IFF=B THENGOSUB1380
1450 PLAY"O2":V=127-Y:U=V/&H2:I=A:S=&H2:Z=A:IFF=B THENGOSUB1380:RETURN
1460 GOSUB1320:IFF=B THEN1380
1470 RETURN
```

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0																												
1							L	A	D	Y	’	S		H	I	G	H	ー	J	U	M	P						
2							K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52	K52						
3																												
4																												
5																			J40									
6																			J20									
7																			J20									
8																			J20									
9																			J20									
10						S	T	A	R	T	!								J20									
11																			J20									
12																			J20									
13																			J20									
14																			J20	M50	M50	M50	M50	M50	M50	M50	M50	
15	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42	G42
16																												
17																												
18																												
19									B	A	R		:						C	M								
20																												

グラム全体の約40%を占めているという事実からも、その複雑さがうかがえる。何しろあまりに複雑なんで作った本人もよくわからないというほどだもんね。だから、ここでは取り上げなかったんだ。実力がある程度ついたときに、解析(かいせき)してみるといいんじゃないかな。きっとキミたちのプログラミングの参考になると思うよ。

⑶うまく動かすのは難しい

レディを走らせるところだけをみたがどうだったかな。ただ走らせるだけでも意外と大変なので驚いたんじゃないかな。でも読むと難しそうだけど、やってみるとそうでもないんだ。考えるよりはやってみたほうがいい。案ずるよりは生むがやすし。とにかくやってみよう。

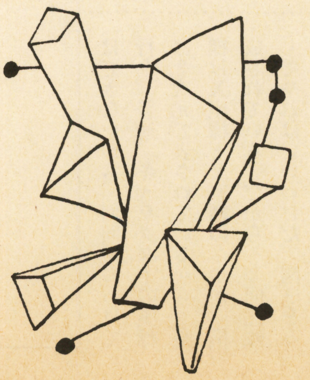
⑷遊び方

キーボードのカーソルキー→とスペースバーを使う。STARTの表示が出たら、とにかく→キーを叩こう。そうするとレディが走り始める。走るスピードは叩くスピードが速いほど速くなる。うまくバーを越えられそうなところでスペースバーを押すと、レディがジャンプする。ジャンプのとちゅうで→キーを押すと、レディの進む方向が少し変更されて、バーを楽に越えられるようになる。以上が1トライで、1つの高さについて3回までトライできる。

クリアの場合は、自動的にバーの高さが高くなり、3回失敗すると今日の最高記録とキミの記録が表示され、GAME OVERだ。

女子の世界記録は2mちょっとだが、レディは軽々とそのくらいはクリアしてしまう。レディはどこまで記録をのばせるのか。その記録をつくるのはキミだ。





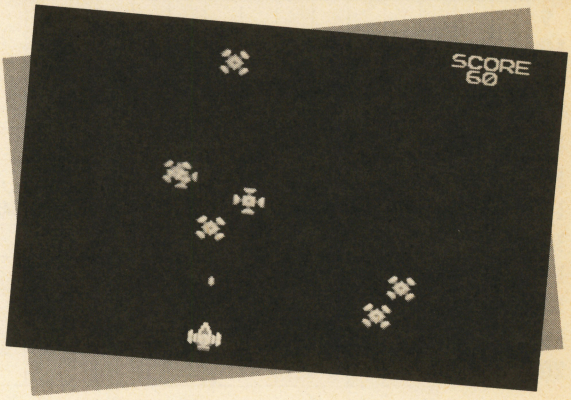
第6章

これがヒットチェック

ヒットチェックで接触シーンもお手のもの

ヒットチェックを効果的に使って、数々のゲームが世の中をさわがせた。これなしではゲームは語ることができない。

しかし、ヒットチェックの基本は、それほど難しいものではない。ここでは、ヒットチェックを理解するうえでぜひ知っておいてほしい大切なポイントについて、解説してみよう。



(1)ヒットチェックをやるべし

敵と自分のキャラクターが接触(せっしょく)したときや、こちらの弾が当たったときのチェックはヒットチェックといわれ、ゲームプログラムにおいて、なくてはならないものだ。基本はそれほど難しいことではないが、よくわかんないって人もいるんじゃないかな。というわけで、ヒットチェックについて勉強しちゃおう。

(2)基礎事項と実例

基本は簡単で、座標を比較してやればいいだけだ。図1を見てほしい。ある1つのキャラクターに注目し、基準点を決定する。この基準点はどう決めてもいいが、ほかのキャラクターも同様にしておこう。図2に、好ましい例とあまり好ましくない例を挙げた。基準キャラクターと比較キャラクターの基準点のとり方が違うと絶対にいけないのではないが、違っているとキャラクターごとにチェックパラメータが変わり、それぞれにチェックルーチンを用意する必要がある。16×16キャラと8×8キャラの比較といった理由で、そうせざるを得ないのならともかく、普通はなんのメリットもないからやめておこう。

基準点が決定したら、あとはその座標を比較するだけだ。一度にはできないから、何回かにわけて行う。普通、キャラクターの座標は2次元で表されているから、座標はX座標とY座標の2つから構成されている。だから、2回比較してやればいいわけだ。図3をみてみよう。このようにX座標とY座標のそれぞれで、キャラクターが重なっているかどうかをみて、

図1

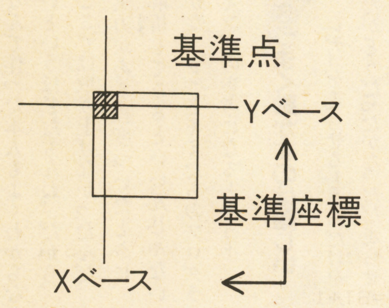
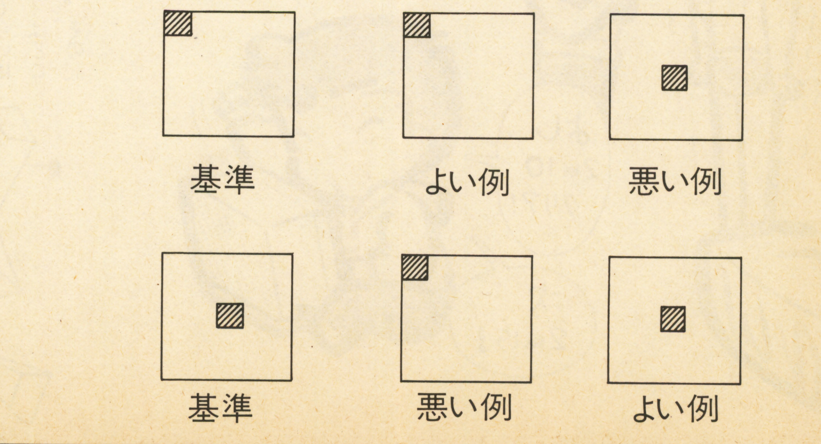
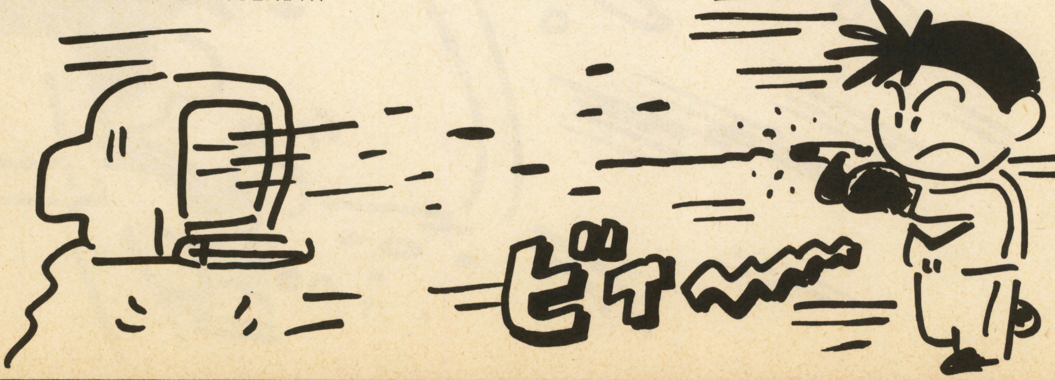


図2



シューティングゲーム

```
10 VIEW
20 FORI=0TO3:E0$=E0$+CHR$(176+I):E1$=E1$+CHR$(180+I):SS$=SS$+CHR$(172+I):NEXT
40 X=115:Y=195:XX=4:LL=16:RL=230:UL=30:DL=235:BY=100:SP=5:RD=1:SB=30:ST=1:LN=89
50 DIMSS(SP)
70 LOCATE22,0:PRINT"SCORE":LOCATE23,1:PRINTSC
100 DEFSPRITE6,(0,1,0,0,0)=SS$:SPRITE6,X,Y:SPRITEON
1000 '
1010 GOSUB7000:S=STICK(0):T=STRIG(0):SPRITE6,X,Y
2010 IF MOVE(7)=-1 THEN3010
2020 ERA7:LX=-8:IF T<8 THEN3010
2025 DEFMOVE(7)=SPRITE(12,1,1,LN,0,0)
2030 LX=X+5:LY=Y-6:POSITION7,LX,LY:MOVE7
3010 FORI=0TOSP:IF ABS(XPOS(I)+4-LX)<8 THEN IF ABS(YPOS(I)+3-YPOS(7))<8 THEN GOS
UB 10000:I=SP
3020 NEXT
4000 IF S=0 THEN5000
4005 ON S GOTO4010,4020
4006 GOTO5000
4010 X=X+XX:IF X>RL THEN X=RL
4015 SPRITE6,X,Y:X=X+XX:IF X>RL THEN X=RL
4016 GOTO5000
4020 X=X-XX:IF X<LL THEN X=LL
4025 SPRITE6,X,Y:X=X-XX:IF X<LL THEN X=LL
5000 '
5005 FORI=0TOSP:IF MOVE(I)=0 THEN5020
5010 IF ABS(Y-YPOS(I))<13 THEN IF ABS(X-XPOS(I))<13 THEN11000
5020 NEXT
6000 GOTO1000
7000 FORI=0TOSP:IF MOVE(I)=-1 THEN7040
7020 SY=RND(Y-BY+RD*8):SX=LL+RND(RL-LL-16):SS(I)=RND(2)+1
7022 SL=(DL-SY)/2:SD=4:IF ABS(X-SX)<(Y-SY)/3 THEN SD=5:GOTO7030
7023 IF SGN(X-SX)=-1 THEN SD=6
7030 DEFMOVE(I)=SPRITE(7,SD,SS(I),SL,0,0):POSITION I,SX,SY:MOVEI
7040 NEXT:RETURN
10000 FORN=0TOSP:IF N<>I THEN CUTN
10010 NEXT:ERAI,7:LX=-8
10020 FORN=0TO9:SWAP E0$,E1$:DEFSPRITEI,(0,1,0,0,0)=E0$:SPRITEI,XPOS(I),YPOS(I):
PAUSE10:PLAY"O5F1":NEXT:SPRITEI
10030 SC=SC+10*(3-SS(I)):LOCATE22,1:PRINTSC:ST=ST+1
10031 IF ST<SB THEN10040
10032 BEEP:ST=1:RD=RD+1:IF RD>5 THEN RD=5
10040 FORN=0TOSP:IF N<>I THEN MOVEN
10050 NEXT:RETURN
11000 '
11010 CUT0,1,2,3,4,5,6,7:SPRITE6
11020 FORI=0TO9:SWAP E0$,E1$:DEFSPRITE7,(0,1,0,0,0)=E0$:SPRITE7,X,Y:PAUSE 10:PLA
Y"O5F3":NEXT:SPRITEOFF:CLS
11030 LOCATE5,10:PRINT"*** GAME OVER ***":LOCATE5,12:PRINT"TRY AGAIN (Y/N)"
11040 K$=INKEY$:IF K$="" THEN11040
11050 IF K$="Y" THEN RUN
11060 END
12000 'PROGRAM VERSION 1.0
12010 'PROGRAMED BY T.UKITA
```



どちらも重なっていたら衝突していると判断する。基本はこれだけだ。簡単なもんだね。

基本がわかったところで、実例をみてみよう。「シューティングゲーム」のプログラムをみてほしい。3010行と、5010行がヒットチェックをしているところだ。3010行は、こちらの撃ったレーザーが敵キャラクター（スピナー）に当たっているかどうかをみていて、5010行ではスピナーとキャラクター（スターシップ）が接触したかどうかをチェックしている。

前者は16×16のキャラクターと8×8のキャラクターのヒットチェックをしていて、後者は16×16のキャラクター同士のヒットチェックをしている。そのため、ちょっとパラメータが違っている。絶対値をとっているのは、上下左右の区別をなくすためだ。また不等号のあとの数字は、衝突しないぎりぎりの距離を表している。3010行の8というのはわかるけど、5010行のほうがなんで16でなくて13なんだと思ったキミは鋭い。これは実は、スプライトの大きさは16×16でも、キャラクターの大きさが16×16じゃないからなんだ。ほとんどのキャラクターは13×13ぐらいで構成されているから、16にしてしまうと見た目は当たったようにみえないのに、当たったと判定されるということが起こってしまうんだ。だから13にしている。実際にやってみるとよくわかるからやってみるといい。

3010行のほうは基準点が違っているから、座標の比較計算の際にパラメータをつけて計算している。だから、8のままだ。これは実際にやってみないとわからないノウハウだ。難しいもんだね。

③ヒットはさらに続く

ヒットチェックについて話してきたけど、これは別にファミリーベーシックにかぎったことじゃないから、別なマシンでゲームを作るときにも応用してほしい。実はまだ少し問題があるが、ここではあえて説明しなかった。とりあえずやってみよう。

④遊び方

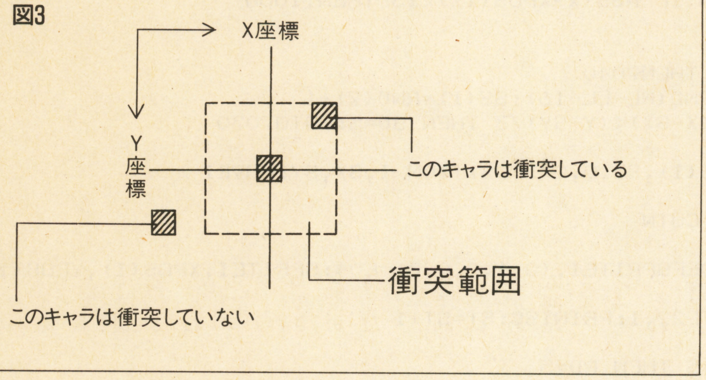
☆ストーリー☆

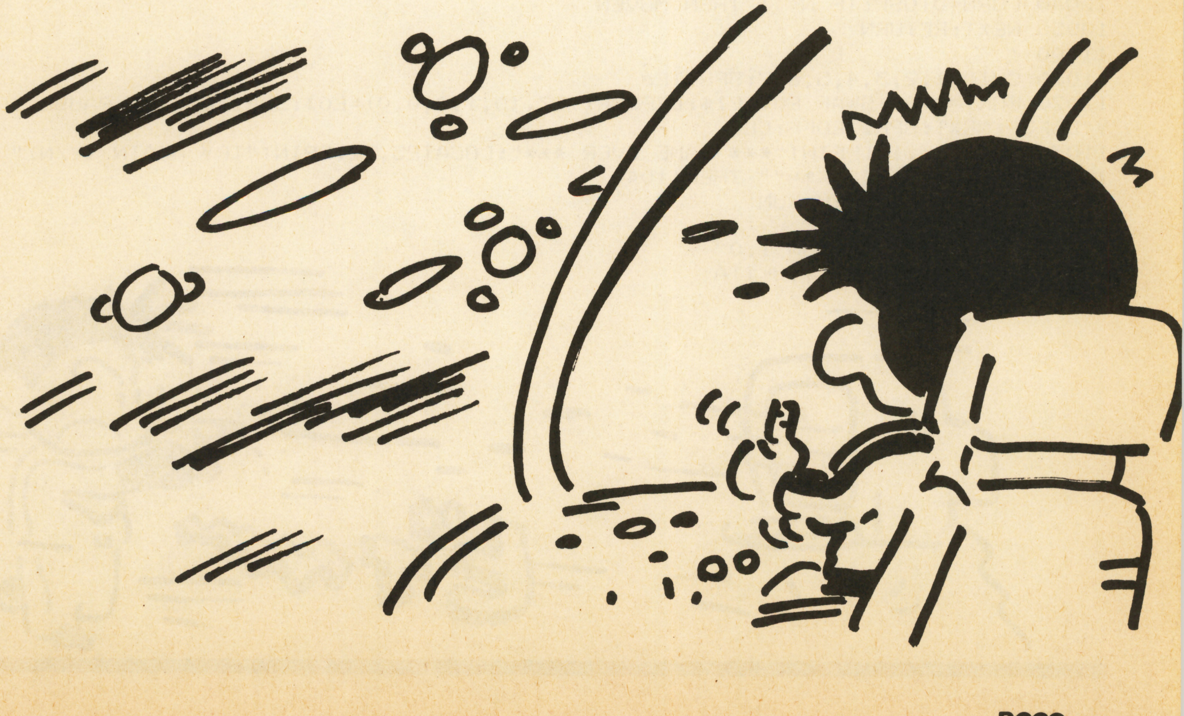
それはあまりにも突然のできごとだった。順調にパトロールを続けていたわれわれの前にそいつらが現れたのは。しかもその謎の一団は、体当たり攻撃をかけてきたのだ。われわれは必死で彼らに呼び掛けた。しかしなんの返事も返ってこない。そこでわれわれは、これを撃破することを決意した。

☆やりかた☆

ゲーム方法はいたって簡単。コントローラーの✢ボタンでパトロール船（スターシップ）を左右に動かし、Aボタンでレーザーを発射して敵（スピナー）を撃墜する。敵は1機10点。30機やっつけると1ラウンド終了で、以後敵の出現位置がパトロール船に近づき難しくなる。パトロール船がやられたらGAME OVERだ。

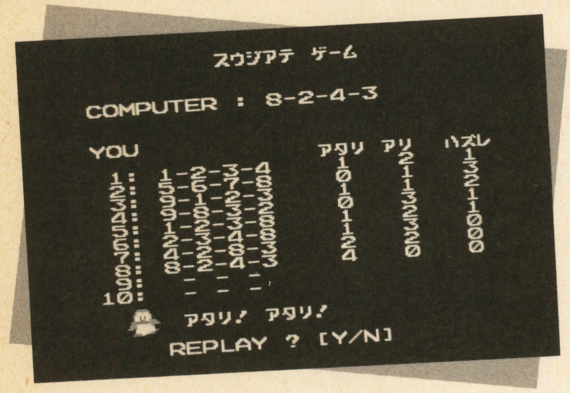
図3





第7章 サウンドで大感激!

音を使いこなしておもしろさ倍増



音楽のないゲームなんて、それはそれはホントーにつまらないもの。もしあのゲームにBGMがついていなかったら、ぜったいにあの感激はえられなかったろうというのは、数々の経験からもみんな感じていることだろう。

このゲームの大親友、サウンドをゲーム作りにとり入れるための勉強をしてみよう。

(1)音楽って楽しいもんだ

ゲームにおけるサウンドの重要性は言うまでもないだろう。ゲームの内容やプログラムがいくら優(すぐ)れていても、サウンドが貧弱だとそのゲームはヒットしないもんね。このため、サウンド専門のプログラマーがいるほどだ。サウンドの重要性は近年ますます高まってきているが、こうした傾向に一役かったのが、サウンド機能を持ったコンピュータの普及だろう。

サウンド機能はいまやホビー向きのパソコンには必ずあるし、ビジネス機にもついているものがあるほどだ。これは、音を作る回路がIC化されたためで、PSG(プログラマブル・サウンド・ジェネレータ)とよばれるICがそれだ。

アメリカのGI社が開発したものがよく使われていて、これは三重和音＋ノイズを同時に出すことができる。しばらく前までは、このタイプのICしかなかったので、どのマシンにもこのICが使われていた。ただ、この音はいかにもコンピュータが作った音って感じがするんだよね。いまいちリアルさや暖かみがなかったんだ。

ところが最近になってこの欠点をなくしたICが登場した。FM音源といわれるのがそれだ。ヤマハが開発したもので、GI社のものとは音を作る原理が違っている。実際に聞いたことのある人も多いんじゃないかと思うけど、本当にいい音を出すごいICだ。ただその代わり、GIのものと比べて設定してやらなければならないパラメータが増え、使い方がちょっと難しくなった。

アーケードゲーム(アタリ社の「マーブルマッドネス」など)にもようやくFM音源が使われだしたところで、まだこれからって感じだけど、どんどん普及していくのは間違いない。そのうちパソコンでも、これをサポートしたすごいソフトがでて、僕たちをうならせてくれるだろう。はやくお目にかかりたいね。

(2)ファミコンでは

さてわがファミコンではどうだろうか。残念ながらファミコンのサウンドジェネレーターはFM音源じゃあないが、ちゃんとPSGが1つ載っている。ファミコンの内部を見るとそれらしい石はないが、カスタム化されているのでわからないだけなんだ。ベーシックでは、PLAY文としてこれをサポートしていて、文字列として、パラメータを与えてやれば音を出してくれる。

そうすると次の問題は、曲をどうやって作るのかってことだ。ところがこいつがとてつもなく難しい問題で、はっきりいって僕のほうが教えてもらいたいくらいだ。



楽器などをやっていた人なら、まずそれで作曲して、その譜面をデータ化すればいい。これが一番確実でよいものができるような気がする。ただ僕もそうだけど、そういうことができない人もいるんだよね。そういう人の場合はどうすればいいかっていうと、どうやら行き当たりばったりにやってみるしかないみたいだ。

画面作成のところでもそうだったけど、やってみるしかないんだね。適当に音を並べてみて、それがBGMとして使えるかどうか

数字をあてろ!

```
10 DIM CN(3),N(3),JU(2)
20 CGSET1,2:C1=4:C2=11:SPRITEON
100 ERA0,1:CLS:VIEW:TC=0
1000 '
1100 FOR I=0 TO 3:CN(I)=-1
1110 X=RND(9)+1:F=0
1120 FOR J=0 TO I:IF CN(J)=X THEN F=1
1130 NEXT:IF F=1 THEN 1110
1140 CN(I)=X:NEXT
2000 '
2010 LOCATE3,22:PRINT "   ｽｳｼﾞ ｦ ｲﾚﾃ ｸﾀﾞｻｲ
2020 TC=TC+1:F=0
2030 FOR I=0 TO 3:N(I)=-1
2040 LOCATE 5+I*2,8+TC:K$=INKEY$(0):K=ASC(K$)-48:IF K$=CHR$(27) THEN F=1:I=3:GOTO2060
2045 IF K<1 OR K>9 THEN2040
2046 FL=0:FOR J=0 TO I-1:IF N(J)=K THEN FL=1:J=I-1
2047 NEXT:IF FL=1 THEN2040
2050 PLAY"T1Y2M0V1504C2":N(I)=K:PRINT RIGHT$(STR$(K),1)
2060 NEXT
2070 IF F=1 THEN TC=10:GOTO5010
3000 '
3010 JU(0)=0:JU(1)=0:JU(2)=0
3020 FOR I=0 TO 3:IF CN(I)=N(I) THEN JU(0)=JU(0)+1
3030 NEXT
3040 FOR I=0 TO 3:F=0
3050 FOR J=0 TO 3:IF I=J THEN 3070
3060 IF N(I)=CN(J) THEN F=1:J=3
3070 NEXT
3080 IF F=1 THEN JU(1)=JU(1)+1
3090 NEXT
3100 JU(2)=4-JU(0)-JU(1)
4000 '
4010 Y=8+TC
4020 LOCATE 15,Y:PRINT JU(0):LOCATE 19,Y:PRINT JU(1):LOCATE 23,Y:PRINT JU(2)
5000 '
5005 IF JU(0)=4 THEN GOSUB 7000:GOTO5100
5010 IF TC=10 THEN GOSUB 6000:GOTO5100
5020 GOTO2000
5100 LOCATE 5,22:PRINT "REPLAY ? [Y/N]
5110 K$=INKEY$:IF K$="" THEN 5110
5120 IF K$="Y" THEN 100
5130 IF K$<>"N" THEN 5110
5200 CLS:SPRITEOFF:LOCATE 5,10:PRINT"***GAME OVER***":END
6000
6005 GOSUB9000
6020 DEFMOVE(0)=SPRITE(C2,7,1,86,0,2):POSITION0,208,178:MOVE0
6030 PD$="ｻﾞﾝﾈﾝﾃﾞｼﾀ!          "
6040 FORX=27TO6STEP-1:LOCATEX,20:PRINT LEFT$(PD$,28-X);
6050 FORD1=1TO103:NEXT:NEXT
6060 CUT0:XX=XPOS(0):YY=YPOS(0):DEFMOVE(0)=SPRITE(C2,0,1,1,0,2):POSITION0,XX,YY:MOVE0
6110 PLAY"T2Y3M1V802B3A3G3F3E3D3C3"
6210 GOSUB 8000
6220 RETURN
7000 '
7010 GOSUB9000
7020 DEFMOVE(1)=SPRITE(C1,7,1,86,0,0):POSITION1,208,178:MOVE1
7030 PD$="ｱﾀﾘ! ｱﾀﾘ!          "
7040 FORX=27TO6STEP-1:LOCATEX,20:PRINT LEFT$(PD$,28-X);
7050 FORD1=1TO103:NEXT:NEXT
7060 CUT1:XX=XPOS(1):YY=YPOS(1):DEFMOVE(1)=SPRITE(C1,0,1,1,0,0):POSITION1,XX,YY:MOVE1
7200 PLAY"T2Y1M1V803G3A3G3A3G8"
7210 GOSUB8000
7220 RETURN
8000 '
8010 FOR I=0 TO 3
8020 LOCATE 12+I*2,4:PRINT RIGHT$(STR$(CN(I)),1):PAUSE 20
8030 NEXT:RETURN
9000 '
9005 LOCATE0,22:PRINT"
9010 RETURN
9900 'PROGRAME VERSION 1.2
9910 'PROGRAMED BY T.UKITA
```

判断するしかない。誰かかんたんに曲が作れるツールでも作ってくれないかなあ。そうすれば、もう少しまともなBGMを出せるんだけど。

⑶PLAY文の使いみち

ちょっと、ぼやきになっちゃったけど、ついでだからもう1つぼやいちゃおう。というのは、このPLAY文には致命的な欠点があるんだよね。使ってみた人はみんなわかっていると思うけど、このPLAY文実行中はMOVE文を除いて他に何ひとつ命令を実行してくれないんだ。つまりアクションゲームなんかで、BGMにPLAY文を使うことは絶対にできないんだ。

ではどういう場合に使うのかというと、画面での動きがMOVE以外になくて、ちょっとウエイトをかけておきたいというようなときに使うんだ。実例を挙げよう。「数字をあてろ！」のリストを見てほしい。PLAY文が使われているところ（2010行、6110行、7200行）は、どれもメッセージを表示した直後のポーズがわりに使っているにすぎない。

これ以外の使い方としては、ごく短い音を連続して出して効果音にするというような使い方（「マリオの走り幅とび」などのスポーツシリーズ）がみられる程度で、本当の意味での使い方（ゲーム途中でのBGM演奏）をしているものは見られない。でもゲームから音を取ってしまったら、とてもゲームにならないのは明らかだ。ためしに「数字をあてろ！」から音を取ってみるといい。それこそ味もそっけもないゲームになってしまう。また他のゲームでも同様のことがいえる。音がないと、ゲームにならない。でもゲーム途中ではBGMを出せない。まったく矛盾している。どうすりゃいいんだろう。困ったもんだ。

⑷サウンドはこれからの課題だね

結局PLAY文は、使えるところを見極めたうえで、使うべきだ。単音発生だけでもうまく使えば立派な効果音となる。メッセージの直後に使えばポーズとBGMの2つを一度に実現できる。ゲーム中でのBGMは、PLAY文ではどうしようもないので、別な方法を使おう。最近KAZ氏によってPOKE文で直接メモリにデータを書きこみ、音を出す方法が発表された。これを使えばゲーム途中でBGMを出すこともできるかもしれない。カートリッジゲームではいいBGMを出しているんだし、きっと何とかなるだろう。

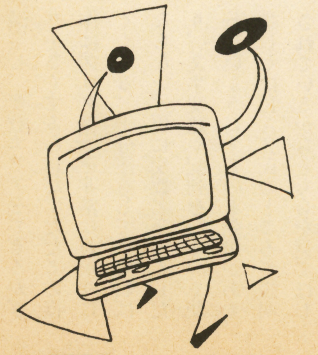
⑸遊び方

1人で遊ぶ。まずファミコンが4つの数字を選び、並べる。もちろんキミには、ファミコンがどういう数字を選び、どう並べたのかまったくわからない。ここからがゲームの始まり。キミは、ファミコンがどういう数字をどう並べたか予想して、4つの数字を順に入力してやる。この入力に対し、入力された数字がファミコンの選んだもののなかにあって、しかも場所もあっているときは、アタリ。場所があっていないときは、アリ。ファミコンの選んだ数字の中に入ってもいないときは、ハズレとしてそれぞれの数をファミコンが教えてくれる。これをヒントにして、この数字はあっているな、こいつはないな、なんて考えていって、ファミコンの選んだ数字とその場所をあててやればいい。数字を入力できる回数は、10回まで。つまり9回までは間違えられる。なお、ファミコンの選ぶ数字は、1から9までで、øはない。また、重複もしないように選んでいる。このことを十分頭に置いて入力する数字を決めないと、とても10回であてることはできない。

BGグラフィック画面データ

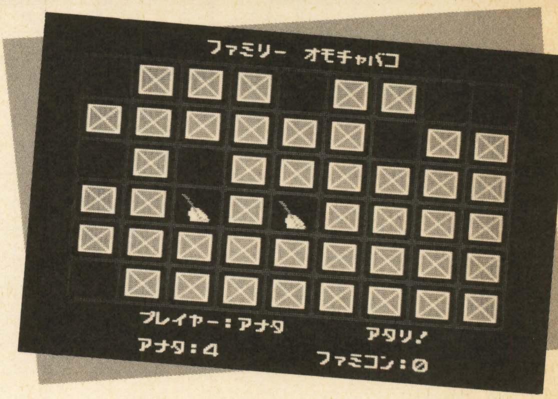
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0																												
1										ス	ウ	ジ	ア	テ		ゲ	ー	ム										
2																												
3																												
4		C	O	M	P	U	T	E	R		:		?	−	?	−	?	−	?									
5																												
6																												
7		Y	O	U																								
8																ア	タ	リ		ア	リ		ハ	ズ	レ			
9			1	:			−		−		−																	
10			2	:			−		−		−																	
11			3	:			−		−		−																	
12			4	:			−		−		−																	
13			5	:			−		−		−																	
14			6	:			−		−		−																	
15			7	:			−		−		−																	
16			8	:			−		−		−																	
17			9	:			−		−		−																	
18		1	ø	:			−		−		−																	
19																												
20																												





第8章
OMエラーをつぶせ！

項目別解消法でエラーとサヨナラ



エラーはキライだ！だいっキライだ！これは、ぜひとも最初にいっておきたい。

ここまでキライなエラーとおサラバするためには、当然アタマだって使ってしまうのだ。

そこで、手始めに相手をOMエラーに定めて、その撃退法の傾向と対策を、みなさんの現在と将来のためにお教えします。

⑴エラーの王者〈OMエラー〉

ファミリーベーシックでゲームを作っていて一番怖いものといえば、やっぱりOMエラーだろう。

プログラムをコーディングして打ちこみだしたら全部入れる前にOMエラーが出たとか、一応プログラムは全部打ちこんだものの、RUNさせたとたんにOMエラーが出たなんてざらだし、ゲームとちゅうで新しい変数を使おうとしたらOMエラーになってしまうこともある。なにしろプログラムと変数の全エリアを合わせても、V2で1982バイト、V3で4086バイトしかないわけで、OMエラーはプログラマーにとって最も怖い存在といえるんじゃないかな。

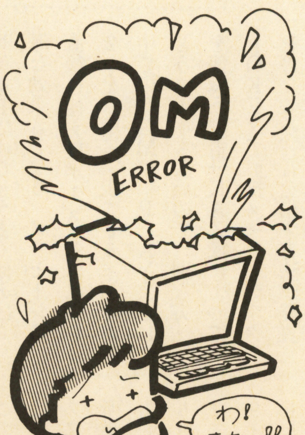
でも逃げているばかりじゃプログラマーの名が泣く。こいつをどう退治すればいいのか考えてみることにしよう。名付けてOMエラー撃退法だ。

⑵撃退法レベル1

まずOMエラーがでる場合から見ていこう。次の2つの場合がある。1つはプログラムを打ちこんでいるときに出る場合。つまりプログラムエリアだけでメモリ不足の場合だ。もう1つは、プログラムは入ったがRUNさせたらOMエラーがでたり、実行とちゅうでOMエラーになる場合で、変数エリアが足りない場合に相当する。

それぞれの場合について撃退法を考えよう。まずは変数エリアが足りない場合から見ていこう。

変数エリアを増やす方法は2つ考えられる。1つはプログラムエリアを減らすというもの。もう1つは、変数を使わないというものだ。具体的にいこう。通常メモリ不足解消のために行われるのは後者のほうからで、変数をできるかぎり使わないようにする。たとえばFOR－NEXT用にいくつもの変数を使っていないだろうか。また、ワークとして1つの変数を共通して使えるところを、全部別な変数にしていないだろうか。FOR－NEXT用変数だって、FOR－NEXTで使っていなければワーク変数として使える。このように考えていって無駄な変数を減らし、変数エリアをできるかぎり小さくするわけだ。ただこの操作はプログラムを非常にわかりにくくする。十分注意してやらねばならない。更に変数名を短くしたり、文字変数や配列をなるべく使わないようにするのも効果がある。詳しくはKAZ氏のところで説明されているからそちらを見てほしい。とにかく変数に関して、いっさい無駄なことをしないようにすればいいんだ。



だいたい以上のようにやれば変数関係のほとんどのOMエラーを撃退できる。しかし、しつこいやつはこれでも撃退できない。そこで次にプログラムエリアの減少化を図ることになる。

⑶撃退法レベル2

プログラムエリアを減らすためには、マルチステートメント化を図ってプログラム行数を減らせばいい。これで、1行あたり行番号分の3バイトぐらいが浮く。でもこの方法はあまり大きな効果はない。大きな効果をあげるには、プログラムの見直しをしたほうがいい。

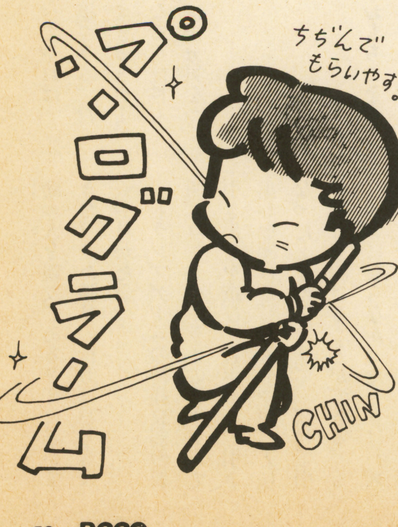
ある処理をするプログラムは、

ファミリーおもちゃ箱

```
1000 A=0:B=1:E=2:G=9:L=5:Q=100:T=3:CGSETB,E:DIMC(53),D(B),H(B),M(12),Z(B):DEFSPR
ITEE,(A,B,A,A,A)=CHR$(214)+CHR$(214)+CHR$(214)+CHR$(214):CLS:LOCATE7,10:PRINT"ｼﾊ
ﾞﾗｸ ｵﾏﾁｸﾀﾞｻｲ":FORI=A TO53
1010 J=RND(27)+B:IFC(J)/Q=E THEN1010
1020 C(I)=C(I)/Q*Q+J:C(J)=C(J)+Q:NEXT:FORI=A TO53:C(I)=C(I)MODQ:NEXT:VIEW:SPRITE
ON:LOCATE5,20:PRINT"ﾌﾟﾚｲﾔｰ:ｱﾅﾀ":LOCATE5,22:PRINT"ｱﾅﾀ:0       ﾌｧﾐｺﾝ:0"
2000 LOCATE19,20:PRINT"     ":S=A:GOSUB3000:GOSUB4000:S=B:GOSUB3000:GOSUB4000:LOC
ATE19,20:IFD(A)=D(B)THEN2050
2010 PRINT"ﾊｽﾞﾚ!":PLAY"T101CBCBC":PAUSEQ:FORW=A TOB:I=(Z(W)-Z(W)/G*G)*T+B:J=Z(W)
/G*T+E:SPRITEW:LOCATEI,J:PRINTCHR$(243);CHR$(247):LOCATEI,J+B:PRINTCHR$(245);CHR
$(248):C(Z(W))=C(Z(W))-Q:PAUSE20:NEXT
2020 FORI=E TOL:M(I-E)=M(I):NEXT:FORI=A TOB:M(L-I)=(Z(I)-Z(I)/G*G)*1000+Z(I)/G*Q
+D(I):NEXT:IFF=A THENF=B:W$="ﾌｧﾐｺﾝ":GOTO2040
2030 F=A:W$="ｱﾅﾀ  "
2040 LOCATE11,20:PRINTW$:GOTO2000
2050 PRINT"ｱﾀﾘ!":PLAY"T102EGEGE":PAUSE50:SPRITEA:SPRITEB:H(F)=H(F)+B:W$=STR$(H(F
)):I=9:IFF=B THENI=22
2060 LOCATEI,22:PRINTRIGHT$(W$,LEN(W$)-B):FORI=A TOL:IFD(A)=M(I)MODQ THENM(I)=A
2065 NEXT:IFH(A)+H(B)<27THEN2000
2070 SPRITEOFF:CLS:LOCATE5,10:PRINT"***GAME OVER***":END
3000 IFF=B THEN3100
3010 SPRITEE,(X+B)*24,(Y*T+5)*8:I=STICK(A):J=STRIG(A):PAUSE5:IFJ=8THEN3080
3020 IFI=B THENX=X+B:IFX>8THENX=A
3030 IFI=E THENX=X-B:IFX<A THENX=8
3040 IFI=4THENY=Y+B:IFY>5THENY=A
3050 IFI=8THENY=Y-B:IFY<A THENY=5
3060 IFJ=E THEN2070
3070 GOTO3010
3080 Z(S)=Y*G+X:IFC(Z(S))/Q<>A THEN3000
3090 C(Z(S))=C(Z(S))+Q:D(S)=C(Z(S))MODQ:RETURN
3100 IFS=B THEN3170
3110 I=B
3120 J=A:FORW=A TOL:IF(M(W)MODQ)=I THENJ=J+B:V=W
3130 NEXT:IFJ=E THENX=M(V)/1000:Y=M(V)/Q-X*10:M(V)=A:GOTO3080
3140 I=I+B:IFI<28THEN3120
3150 W=RND(54):IF(C(W)/Q)<>A THEN3150
3160 Y=W/G:X=W-Y*G:GOTO3080
3170 I=A:FORJ=A TOL:IFD(A)=M(J)MODQ THENW=J:I=B:J=L
3180 NEXT:IFI=A THEN3150
3190 X=M(W)/1000:Y=M(W)/Q-X*10:M(W)=A:GOTO3080
4000 RESTORE4010:FORW=B TOD(S):READI,J:NEXT:DEFSPRITES,(J,B,A,A,A)=CHR$(I)+CHR$(
I+B)+CHR$(I+E)+CHR$(I+T):LOCATEX*T+B,Y*T+E:PRINT"  ":LOCATEX*T+B,Y*T+T:PRINT"  "
:SPRITEE:SPRITES,(X+B)*24,(Y*T+5)*8:RETURN
4010 DATA0,2,12,2,20,2,28,2,40,2,48,2,56,1,60,1,64,2,84,2,88,1,92,1,104,0,108,0,
112,3,116,3,136,2,148,0,160,1,172,2,184,3,188,3,192,2,196,2,200,0,204,0,224,1
```

1つだけではなく、いろいろあるということはわかるね。そしてその中には、現在のものよりも大きなものも、小さなものもあるはずだ。つまりうまく小さくなるものを見つけて、それを今あるものと入れ替えてやればいい。

通常プログラムは処理ごとにまとまっているから、それをチェックする。普通はそう劇的な変化は生じないが、たまにドドッとメモリ節約ができることもある。



マルチステートメント化よりも減る可能性は大きいし、勉強にもなるから試してみる価値は大きい。

⑷撃退法レベル3

ここまで述べてきたことを行えば変数エリア不足によるOMエラーのほとんどが解決するはずだが、それでも撃退できない場合がある。またプログラムエリア不足によるものも撃退できない。ではこれらはどうやって撃退するか。次はこいつを考えよう。

今まで考えてきた撃退法は、プログラムの持つ機能に対して手を加えるものではなかった。しかし、ここまできてしまった以上、プログラムの機能に手を加えることにする。言い換えればプログラムの仕様を変更するのにほかならない。

具体的にいこう。まずメッセージとサウンドの簡略化を行う。

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0									フ	ァ	ミ	リ	ー		オ	モ	チ	ャ	バ	コ								
1	K72	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	K22	K52	K52	L02
2	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
3	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62
4	K42	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K32
5	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
6	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62
7	K42	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K32
8	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
9	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62
10	K42	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K32
11	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
12	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62
13	K42	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K32
14	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
15	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62
16	K42	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K02	K52	K52	K32
17	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62	L32	L72	K62
18	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	K62	L52	M02	L62	L52	M02	K62	L52	M02	K62	L52	M02	K62
19	L12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	K12	K52	K52	L22
20																												

どちらもあればあるほど、ゲームは盛り上がるが、それを必要最小限まで削る。これは、意外と効果が大きい。

一番最初にでるメッセージをBGグラフィック面に書きこんでおくだけでも違う。例えば「ファミリーおもちゃ箱」。BG面データのところを見ると、「ファミリー オモチャバコ」とメッセージが入っている。これをプログラムで出すと、とたんにOMエラーになってしまうから、BG面のほうにもっていった。これと同じようなことは「マリオの走り幅とび」でも行われている。結構メモリが浮くんだよね。

もっと大きい変更をすることもある。例えば「21」の役だ。このゲームの母体であるブラックジャックには、役がいっぱいあって、それぞれで払い戻し倍率が違うところに面白味がある。しかしファミコンではメモリの関係上、すべての役を実現することは不可能なので、思い切って削った。それで役の種類は少ないが、V2でもゲームができるようになったんだ。

このように本来あるべき機能を大幅に削ったり変更したりするわけだ。でもこれはプログラムに直接響いてくることだから十分な検討と注意が必要だ。結局、今まで説明したことをすべてやれば、必ずOMエラーは撃退できる。ただ、あまりにもやりすぎるとゲームとして成立しなくなる可能性もあるから使用の際には十分気をつけてほしい。

⑤遊び方

☆ストーリー☆

ワタル君はいつもおもちゃをペアで買ってファミリーおもちゃ箱にしまっていたが、フトしたことから、おもちゃ箱のファミコンとおもちゃ神経衰弱をすることになってしまった。相手がファミコンという強敵なので、ワタル君は助っ人を親友であるキミに求めてきた。さあ勝負の行方はどちらに。

☆やりかた☆

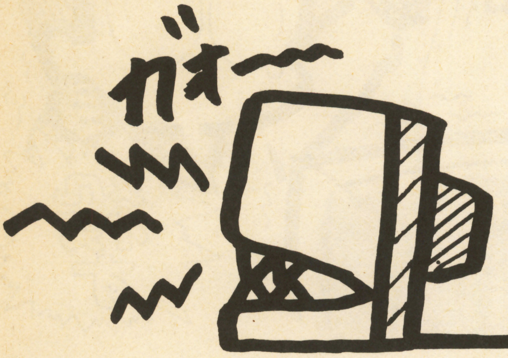
トランプゲームの神経衰弱の要領。コントローラーの十ボタンで引き出しを選び、Aボタンで引き出しを開ける。おもちゃがペアになったときはアタリで、続けて引き出しを開けることができる。もしペアにならなかったときはハズレで、引き出しを開ける権利が相手に移る。これを繰り返して27個すべてのおもちゃがペアになったらゲームオーバー。もし途中でゲームをやめるときは、キミの番にSELECTボタンを押せばよい。



中級編

第9章
一生懸命バックアップ

バックアップをしないと、こんなに恐ろしいことが



ゲームのプログラム作成にはげむ人々に送る心からの忠告。バックアップをやりなさい！
苦労して作り上げたプログラムを、魔の手から守るため、これだけは絶対に必要なことなのだ。
中級編の最後の章として、ここでは著者の苦い経験などを通してバックアップの大切さを伝えようと思う。

(1) バックアップは絶対必要

中級編の最後は、直接プログラムを作るのに役立つものではないが、プログラム製作上、とても重要なバックアップについて話そう。

バックアップなんて聞き飽きたよという人もいるかもしれない。まあ、普通はそれほどバックアップって騒ぐ必要はないんだけど、プログラムを順調に開発するには絶対必要だ。どうしてなのか。この後を読んでもらいたい。

(2) バックアップしないと…

バックアップをしないとどうなるか、僕の恐ろしい経験だ。

経験1。3年ほど前のこと。僕はアルバイトで、あるパソコン上で動く在庫管理プログラムの一部を作っていた。その日、ある程度プログラムが完成したので、ちゃんと動くかどうかチェックしてみた。ちなみにこのプログラムは、ディスク上にファイルを作り、そこにさまざまなデータを書きこむというものだ。ディスクにプログラムをセーブしたのち、僕はプログラムを走らせた。案の定バグがあった。そこでSTOPキーを押してプログラムの実行を止め、プログラムを修正した。そしてまたプログラムをセーブしてその日の作業を終えた。

次の日、昨日セーブしたプログラムを読み出してみると、プログラムのほとんどがなくなっていた。原因は、STOPキーを押した後でファイルをクローズせずに、プログラムをセーブしたからだ。まあ、この程度のことなら仕方ないですむ。そう、バックアップがあれば。ところが僕はバックアップを取っていなかったんだ。おかげで、それから数日は散らばったプログラムの回収に費やされることになってしまった。まったくひどい目にあったもんだ。

経験2。これはファミコンであった。そのとき僕はV3カートリッジで、あるゲームを作っていたんだけど、夜もだいぶ更けてきたんで、その日はそこでやめることにした。ファミリーベーシックのカートリッジには、バッテリーバックアップのスイッチがついていて、スイッチONでベーシックプ



ログラムを残すことができるようになっている。そこで、その日もスイッチをいれてバックアップしようとしたら、いきなり画面が灰色になってしまったんだ。ま、まさかと思いながら、バックアップのスイッチをいれ、1度電源を切ったのち、また電源をいれてみたところ…ものの見事にプログラムはなくなっていた。眠気なんていっぺんで吹き飛んでしまったね。

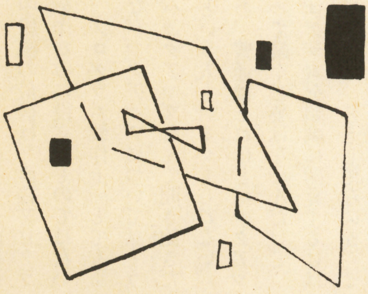
原因はV3カートリッジとファミコン本体の間の接触不良だった。約3キロほどのプログラムだったけど、えらく苦労しただけに、ものすごく腹が立った。バックアップを怠った自分に対してだ。テープにセーブさえしておけば…悔やんでも悔やみきれない結末を迎えたのだった。結局このゲームプログラムは永遠に失われてしまった。

経験3。もう1つファミコンで経験した例を挙げよう。これもV3カートリッジで起こったものだ。やっぱり、ゲームを作っていたんだけど、CHR$を打ちこもうとしてついうっかりファンクションキーの4番を押してしまった。V2ではCHR$だが、V3ではそのキーはGAME3となっているんだ。おかげでいきなりROM中のゲームが呼び出されて、僕が作っていたプログラムは跡形もなく消えてしまったんだ。そしてこれまた当然のごとくテープにバックアップしていなかったもんで、また一から作り直すことになってしまった。もう哀れの一言に尽きたね。



③レッツ！バックアップ

もう賢明な読者諸君ならバックアップがどんなに大切なものかわかっただろう。プログラムの復活ぐらい面白くない作業はないからね。とにかくいつ何時に事故が起こるかわからない。順調にプログラムを作ろうと思ったら、バックアップをするべきだ。





上級編

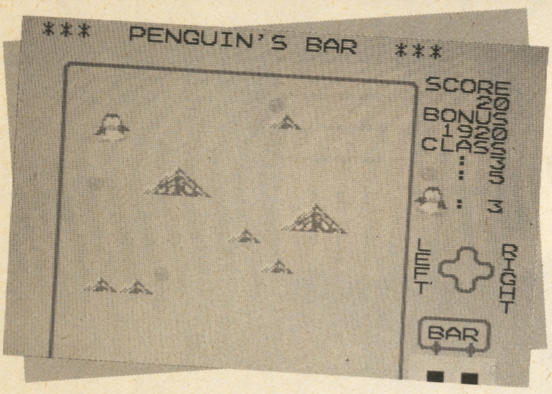
ファミリーベーシックのしくみを
知って、より力をつけよう！



上級編

第1章
MOVE文で動きだす!

アニメーションで楽しもう



さて、ここからは上級編だ。もっと深くファミリーベーシックを掘り下げ、チョットひとあじ違った使い方にチャレンジしてみよう。
　ではまず、キャラクターをアニメーションで動かせるMOVE文について、その動作の原理と、使い方について、より詳しく見ていこう。

(1) MOVE文の使い方

MOVE文はキャラクターテーブルにある16種類のアニメキャラクターを、アニメーションで移動させることのできる命令だ。いくつかの条件を与えてやるだけで、マリオやレディは手をふって歩き出し、自動車はタイヤを回して走り出す。表示するキャラクターのコードなどを気にしなくても、簡単にアニメーションを行えてしまうのが、この命令の魅力だ。
　与えてやる条件は、DEFMOVE文では、重なったときの優先度の番号、キャラクターの種類、動作方向、動きの速さと全移動量、背景の前に表示するか後にするか、そして、配色番号を指定する。次にPOSITION文で、動かしたいキャラクターのスタート位置を決めて、MOVE文で移動開始となる。さらに、移動中のキャラクターを停止させるCUT文。キャラクターを画面上から消してしまうERA文など、動きもいろいろ制御できる。
　移動の情報も、移動中を知らせるMOVE関数、位置を知らせるXPOS、YPOSがある。

(2) MOVE文の動き方と利用法

MOVE文で動くアニメーションは、ベーシックをSTOPキーなどで止めても働きつづけている。
　これはMOVEの動きが、ベーシックの動作と独立して行われていることを示している。つまり、ファミリーコンピュータは、一定の時間ごとに、ベーシックの動作を止めて、キャラクターを動かす処理を行っている。これを割り込み処理という。MOVE文を動かす処理をしているわけだからベーシックの速度も少し遅(おそ)くなってしまう。PLAY文を使ってみると、特にその差がわかると思う。この特徴を利用すると、時間制限のタイマーとして使ったり、ゲームの本体以外の動きを表現できる。

(3) 入力法

ファミリーベーシックのバックアップスイッチをOFFにした状態で電源を入れることを、「コールドスタート」という。プログラムが何も入っていない状態のことだ。



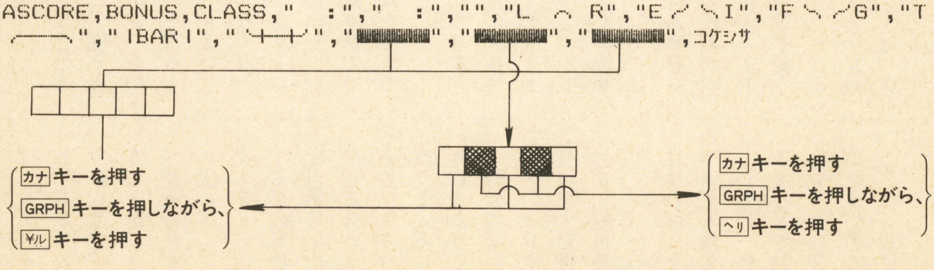
リストを打ちこんで、カセットにセーブ。もう一度、リストを確認してから実行しよう。

(4) 遊び方

一日のつらい仕事を終えて、ホッと一息のペンジン君。今日も、シメキリやぶりのKのためにさんざんの1日だ。帰りにチョット一杯ひっかけていこうと思ったとたんに、スッテンコロリン。お金をばらまいてしまった。疲れで足もとはフラフラ。早くお金を集めないと、酒場が閉まってしまう。ペンジン君をあやつってお金を集めてくれ。+ボタンの右、左でペンジンの歩く方向を左右に変えられる。氷山にぶつからないようにコインをみんな集めて、ペンギンズバーへ直行だ。制限時間内にクリアすると、次の面。お金がふえたり、ペンジンの足が速くなったりだんだんシビアになるぞ。

ペンジンのペンギンズバー

```
6000 Z=0:Y=1:X=2:P=Z:CGSETY,Z:PALETS Y,60,53,48,17:PALETB Y,60,48,48,17:PALETB X
,60,48,54,54:PALETS Z,60,48,53,18:A$="────────────────────"
6050 Q=6:G=Z
6100 C=3
6200 F=5:K$=CHR$(207)
6300 CLS:PALETB Z,60,47,48,17:ERAX:SPRITEOFF:PRINT"***  PENGUIN'S BAR  ***":PRIN
T:PRINT"  ╱";A$;"╲":FORA=3TO22:PRINT"  |":LOCATE21,A:PRINT"|":NEXT:PRINT"  ╲";A$;"╱
";
6400 FORA=Y TORND(Q)+Q:I=X*RND(8)+X:J=X*RND(8)+5:COLORI,J,Y:COLORI+X,J,Y:LOCATEI
+Y,J:PRINTCHR$(200)+CHR$(201);:IFRND(10)>4THENLOCATEI,J+Y:PRINTCHR$(200);CHR$(20
2);CHR$(203);CHR$(201);
6500 NEXT:RESTORE10000:A=X
6550 READB$:LOCATE22,A:PRINTB$;:A=A+X:IFA<12THEN6550
6600 READB$:IFA<24THENLOCATE22,A:PRINTB$;:A=A+Y:GOTO6600
6650 LOCATE27-LEN(STR$(G)),3:PRINTG;:LOCATE23,5:PRINT"2000":LOCATE22,8:PRINTK$;"
 :";F:COLOR22,8,X:LOCATE25,10:PRINTP:DEFSPRITEZ,(Z,Y,Z,Y,Z)=B$:SPRITEZ,190,98:SP
RITEON
6680 LOCATE24,7:PRINT":";C:IFP THEN7000
6700 P=3:LOCATE3,4:PRINT"PUSH START BOTTUN":I=7
6730 IFI=3THENI=7:J=176:A=Y:GOTO6750
6740 I=3:J=24:A=Z
6750 DEFMOVE(X)=SPRITE(4,I,Y,74,A,Z):POSITIONX,J,48:MOVEX
6800 IFSTRIG(Z)=1THEN6050
6850 IFMOVE(X)THEN6800
6900 GOTO6730
7000 E=0:L=2000:FORA=Y TOF
7100 I=RND(15)+3:J=RND(18)+X:IFSCR$(I,J)<>" "ORSCR$(I,J,Y)=CHR$(Y)THEN7100
7200 COLORI,J,X:LOCATEI,J:PRINTK$;:NEXT:I=30:J=40:D=2:PLAY"T1YOM1V604"
7300 DEFMOVE(X)=SPRITE(4,D+Y,C,X,Z,Z):POSITIONX,I,J:MOVEX
7400 PLAY"C1"
7500 K=STICK(Z)AND3
7530 IFK=X THEND=D-Y:IFD<Y THEND=7
7560 IFK=Y THEND=D+Y:IFD>7THEND=Z
7600 PLAY"G1"
7700 I=XPOS(X):J=YPOS(X):M=(I-8)/8:N=(J-14)/8:B$=SCR$(M,N):IFB$=" "THEN8200
7800 IFB$=K$THEN7900
7850 ERAX:COLORM-Y,N-Y,Y:COLORM+Y,N-Y,Y:COLORM-Y,N,Y:COLORM+Y,N,Y:LOCATEM,N:PRIN
TCHR$(219);:PLAY"O2G8":LOCATEM-Y,N:B$=CHR$(216)+CHR$(217)+CHR$(218):PRINTB$;:PLA
Y"G6":LOCATEM-Y,N-Y:PRINTB$;:LOCATEM-Y,N:PRINT" ";CHR$(213);" ";:PLAY"G904":P=P-
Y:GOTO6300
7900 G=G+10:LOCATE27-LEN(STR$(G)),3:PRINTG;:LOCATEM,N:PRINT" ";:PLAY"O3G1C104":E
=E+Y
8000 IFE=F THENLOCATE21,22:PRINTK$;:GOTO8200
8100 IFE>F THEN8300
8200 L=L-10:LOCATE27-LEN(STR$(L)),5:PRINTL;:GOTO7300
8300 G=G+L:LOCATE27-LEN(STR$(G)),3:PRINTG;:PLAY"T1":PLAY"O3C5C4O2G5G4A5A4E5E4G5E
4D5D4C9"
8400 F=F+X:IFF<11THEN6300
8500 C=C-1:IFC>Z THEN6200
8600 Q=Q+1:GOTO 6100
10000 DATASCORE,BONUS,CLASS,"  :","  :","","L ⌒ R","E╱ ╲I","F╲ ╱G","T ╲╱ H","
      T","╱──╲","|BAR|","╲┼─┼╱","██████████","██████████","██████████",コケシサ
```

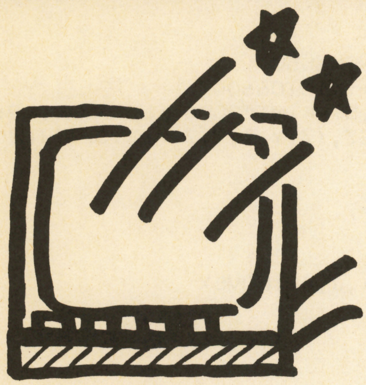




上級編

第2章 カラーリングでワクワク!

色を使ってきれいに画面演出



ファミリーコンピュータは、52色の中から任意の色を選んで、1つのキャラクターに3つの色をつけることができる。これが色とりどりの美しい画面を作り出しているわけだ。

この章では、ファミリーベーシックの色の表現方法とその特徴について述べ、色付けのためのパレットツールを付録とした。



(1)ファミコンの色出し機能について

一般に画面に色をつける方法でよく使われるのは、画面を構成する画素の1ドットずつに色の情報を持たせる方法だ。しかし、この方法には、多くのメモリ容量が必要だ。それに、制御するにはある程度の時間が必要で、高速な処理をするゲームには向かなくなる。

ファミリーベーシックでは、表示できるキャラクターを、テーブルA、テーブルBに示されたものだけにかぎり、表示に関する省メモリ、高速化を実現している。キャラクターに対する着色についても工夫されている。キャラクター内で目や手足、服装、背景用のキャラクターでは、擬似グラフィックのパターンを、3色の色づかいで区別できるように最初から決めてしまい、あとで、その3種類の色の組み合わせを52色の中から選んで使用できるようにしてある。こうすれば、色を最初に決めてやるだけで、あとはそのキャラクターの色について気にしなくてよいし、同じキャラクターの色を変えて区別をつけたりもできるわけだ。

(2)ベーシックでの色のつけ方

では、具体的にベーシックプログラムで色をつけていく手順を説明しよう。

確かに52色内から好きな色を選ぶことができる。しかし、色を選ぶプログラムを必ずゲームの最初に組まなくてはならないとなると、かえってメモリを使ってしまい、ゲーム本体を組む場所がなくなってしまう。そこで、ファミリーコンピュータには、すでにいくつかの色の組み合わせが登録されていて、かんたんな命令で選択できるようになっている。

CGSET命令がそれである。実際の色の組み合わせは、マニュアルの最後にあるカラーチャートを参照しよう。なかに少しコードの間違いがあったので正しい表を載せておいた。CGSET文では、まず背景用の色の組み合わせを0か1で指定する。それぞれのグループでは、3色1組の色のコードが4組、計12色の色を背景で使うことができる。

BGグラフィックスで作った画面と同じ色で使いたいときは、プログラム中で、1のパレットコードをCGENの背景用として選んでやればよい。

次に、CGEN文で指定するのは、スプライト用のパレットコードである。これは0~2の3組の中から選べ、各12色の色を表現することができる。こうして画面内で使える色が決定されたわけだ。

次にベーシックでは、キャラクターをどの色の組み合わせで表示するかの処理を決める。表にある配色番号というのが、その横の色の組み合わせにつけられた番号である。1キャラクター内の定められたところにそれぞれの色が設定されるようになっている。

背景の場合は、BGグラフィックスのセレクトモードでRETURNキーを押すごとに変わる色番号が、配色番号である。CGSETで背景色のパレットコードを1にすれば、そのままの色がベーシックでも使える。

背景の一部や、新しくPRINT文などで加えた部分の色を変えたいときに使用するのが、COLOR文である。この命令は残念なことに、背景1キャラクターずつ



色出し活用パレットツール

```
5 DIM P(7,3):CGSET1,0:SPRITEON:VIEW
10 DATA48,33,2,48,39,24,48,39,22,41,54,23
20 DATA54,22,2,39,48,25,53,37,23,48,39,22
30 FORA=0TO7:READP(A,1),P(A,2),P(A,3):P(A,0)=31:NEXT
40 B=0:C=0:D=4:F=0
50 FORA=0TO3:DEFSPRITEA,(A,D,B,B,B)=CHR$(B)+CHR$(1)+CHR$(2)+CHR$(3):SPRITEA,80,3
9+24*A
52 DEFSPRITEA+4,(A,D,B,B,B)=CHR$(214)+CHR$(215)+CHR$(216)+CHR$(240+A):SPRITEA+4,
104,39+24*A
55 GOSUB 1200
60 NEXT
100 FOR A=0TO63:LOCATEA/16*3+15,(A AND15):PRINTA;:NEXT
110 LOCATE 16,17:PRINT"BG COLOR";
120 GOSUB900:LOCATE0,0:PRINT"C";
200 GOSUB2000:K=ASC(INKEY$):IFK=0THEN200
210 IFK=13THENF=(F+4)AND255:FORA=0TO3:POKE&H680+A*4,F,F+1,F+2,F+3:SPRITEA,80,39+
24*A:NEXT
220 IFK>=48ANDK<=51THEN500
230 IFK=66THENB=0:D=0:LOCATE0,0:PRINT"B";
240 IFK=67THENC=0:D=4:LOCATE0,0:PRINT"C";
250 GOTO200
290 GOTO200
500 GOSUB2010:IFD THEN1000
505 B=K-48:GOSUB900:LOCATE19,19:PRINT"*";
510 B=K-48:FORU=0TO3:W=P(B,U):GOSUB800:GOSUB900:NEXT:LOCATE19,19:PRINT" ";:GOTO2
00
800 Z=W:GOSUB1500
805 IFD THENP(D+C,U)=W:PALETS C,P(D+C,0),P(D+C,1),P(D+C,2),P(D+C,3):GOTO810
807 P(B,U)=W:PALETB B,P(B,0),P(B,1),P(B,2),P(B,3)
810 K=ASC(INKEY$)-27:IFK<1ORK>5THEN800
820 ONK GOTO830,840,850,860,895
830 W=W+16:GOTO870
840 W=W-16:GOTO870
850 W=W-1:GOTO870
860 W=W+1 '
870 W=W AND63
880 GOSUB 1510
890 GOTO800
895 GOSUB1510:RETURN
900 COLOR20,19,B:LOCATE20,19:PRINTCHR$(253);CHR$(254);:LOCATE20,20:PRINTCHR$(255
);
910 LOCATE22,19:PRINTB;:LOCATE14,22:PRINTP(B,0);P(B,1);P(B,2);P(B,3);:RETURN
999 LOCATE 0,0
1000 C=K-48:LOCATE7,3+3*C:PRINT"*";:FORU=0TO3:W=P(D+C,U):GOSUB800:A=C:GOSUB1200:
NEXT:LOCATE7,3+3*C:PRINT" ";:GOTO200
1200 E=A+4:LOCATE12,4+3*A:PRINT"    ";:LOCATE0,4+3*A:PRINTA;":";P(E,0);P(E,1);P(E
,2);P(E,3);:RETURN
1500 X=Z/16*3+15:Y=Z AND15:LOCATEX,Y:PRINT">";:RETURN
1510 X=Z/16*3+15:Y=Z AND15:LOCATEX,Y:PRINT" ";:RETURN
2000 LOCATE0,23:PRINT"RETURN,C,B,1,2,3,4";:RETURN
2010 LOCATE0,23:PRINT"CURSOR MOVE        ";:RETURN
```

の色を変えることはできない。2×2の4文字分を1まとめとして、そのなかの色すべてを変えてしまうのである。指定方法は、横、縦の位置と配色番号である。横方向は2文字ずつ、縦方向は、上、下1列ずつが1文字、あとが2文字ずつ、同じエリアに含まれている。

スプライトのキャラクターに色付けする方法は、DEFSPRITE、DEFMOVEでキャラクターを指定するとき、配色番号を指定することで行う。それによって、キャラクターの指定の位置に3色の色がつけられることになる。たとえばマリオの顔と手、耳は一番左端の色、目、ひげ、ズボンは右端の色、帽子、上衣、クツは中央の色がそれぞれ着色される。

さて、オーダーメードの色がどうしても気にいらないという人のために用意されているのが、PALET文だ。S、Bでスプライトか背景かを選び、配色番号を決める。そして、画面全体の色(配置番号0のときのみ指定できる)、左端の色、中央の色、そして右端の色を、コードで指定することができる。自分好みの色の組み合わせを探してみよう。

パレットコード対応表

スプライト用

	配色番号			
パレットコード0	0	54	22	2
		36	16	02
	1	39	48	25
		27	30	19
	2	53	37	23
		35	25	17
	3	48	39	22
		30	27	16
パレットコード1	0	48	22	1
		30	16	01
	1	16	0	1
		10	00	01
	2	48	41	9
		30	29	09
	3	48	22	7
		30	16	07
パレットコード2	0	48	38	18
		30	16	12
	1	48	21	18
		30	15	12
	2	48	18	22
		30	12	16
	3	48	38	25
		30	26	19

バックグラウンド用

	配色番号			
なし	0	48	33	2
		30	21	02
	1	48	39	24
		30	27	18
	2	48	39	22
		30	27	16
	3	41	54	23
		29	36	17
パレットコード0	0	44	21	7
		2C	15	07
	1	39	33	18
		27	21	12
	2	41	54	23
		29	36	17
	3	48	38	7
		30	26	07
パレットコード1	0	48	33	2
		30	21	02
	1	48	39	24
		30	27	18
	2	48	39	22
		30	27	16
	3	41	54	23
		29	36	17

▲これが、パレットカラーの初期設定値だ。スプライトのパレットでは、CGSETを行わないと、自動的に、バックグラウンドのパレットコード1が選択される。表には「なし」としておいた。

⑶パレットツール使用法

自分で作った背景に、キャラクターを置いて、色をいろいろつけてみるツールだ。まずリストを打ちこんで、セーブしておこう。

次にBGグラフィックで画面の左半分に自由な背景を描いてみる。

そしてRUN⏎。キーボードからの操作で好みの色を選ぼう。

●操作法

C、B…画面左上端にスプライトを色づけするときはC、背景を色づけするときはBと出力される。

Ø～3…配色番号を指定し、それに対応するグループの色を変えられる。

↑→↓←…画面右の色コード表内の〉印を動かす。その色がつけられる。

スペースバー…配色番号内の次の色コードを指定する。

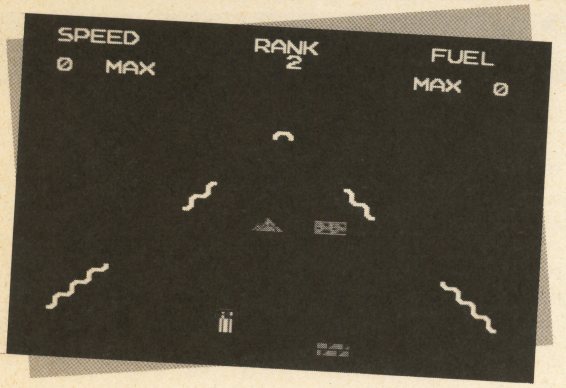
リターンキー…スプライトのキャラクターの種類を次々と変える。キャラクターテーブルAのキャラクターを表示する。

以上が色づけに必要なキーボード操作である。表示については、画面右下に背景コードの配色が四角に示され、コードが横に表示される。スプライトの色は、キャラクターの右横に四角で示され、コードが下に表示される。自分の好みの色がみつかったら、そのコードを別に書きとめておいて、プログラム中で、スプライトならPALETSで、背景ならPALETBで設定してやればよい。



第3章 いろいろ楽しいPALET文

もっと色であそんでしまおう



第2章に引きつづき、この章でも色についてのお勉強。パレット機能の原理や利点、効果的な使い方などを見ていきたい。ただ色をつけるだけではなく、背景色との組み合わせで、動きを出すテクニックなども合わせて説明しよう。

(1) パラパラマンガはなぜ動く

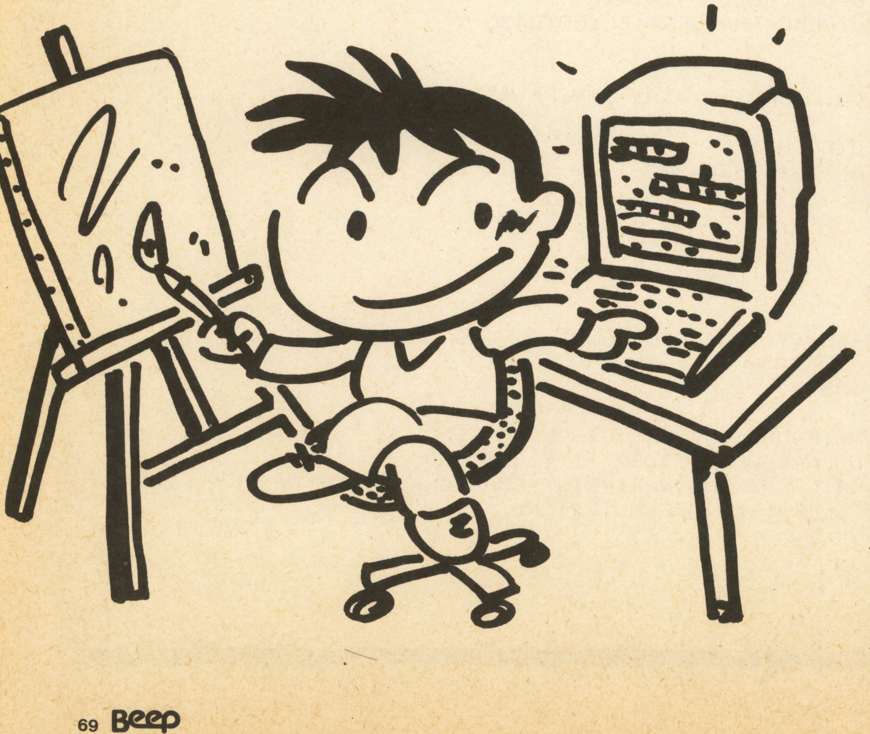
ぼくはむかし授業がヒマになると、ノートの片すみにかんたんな線画のアニメーションを描いて、パラパラめくって動きを楽しんだんだ。キミには経験ないかな？　けっこうラフに描いても、ちゃんと動いて見えるから不思議なものだ。その気になって見てみると、とびとびのあらい動きでも、連続した動きに見えてくる。これは、人間が目で見たものを理解するとき、それまでの経験などをもとに、連続した動きとして受けとってしまうからである。

この効果を逆に利用しようというのがこの章のネタである。人間の思いこみを利用しようというわけだ。

(2) パレットの考え方について

パレット。そう、絵を描くときに、絵の具をしぼり出し、ゴチャゴチャとかきまぜる板のことである。この名がコンピュータの命令についているのは、画面に色をつけるときの中継役を、この命令が受け持っているからである。つまり、画面に絵を描くときに、直接の色を指定するのではなく、パレットコードという番号で指定するのだ。パレットのこの部分にある絵の具で、この部分を塗りますよと決めてしまうのだ。そして、パレットに何色の絵の具を出すかによって、画面の色を決定する。

この機能をもつと、どんな利点があるのだろうか？　それには、まず、画面の絵と本物の絵の差について考えてみることが必要だ。本物の絵は、パレットに出された絵の具で色を塗ってしまえばそれまでである。塗り直すこともできるが、汚くなる一方だ。しかし、画面の絵は違う。パレットにある色のコードを変えてやれば、即座にそのパレットコードで描かれた部分すべてが変わるのである。これは重要な機能だ。



(3) パレットで動きを表現。

さて、パラパラマンガと、パレットの機能がどう結びつくかというと、ファミリーベーシックでは、

背景、スプライトともに4組ずつの配色番号の組み合わせを使うことができる。そして、それぞれをPALET文で好きな色に指定できる。例えば、それを画面全体の色と同じ色に指定すると、そのキャラクターは見えなくなってしまうわけだ。ゲーム作りで臨場感を出すために、流れる星や道路は、なくてはならないものである。しかし、その処理に時間がかかるようでは、ゲーム全体がスピードダウンしてしまい、逆効果だ。人間の目は多少あらい動きでも連続した動きとして見てくれる。

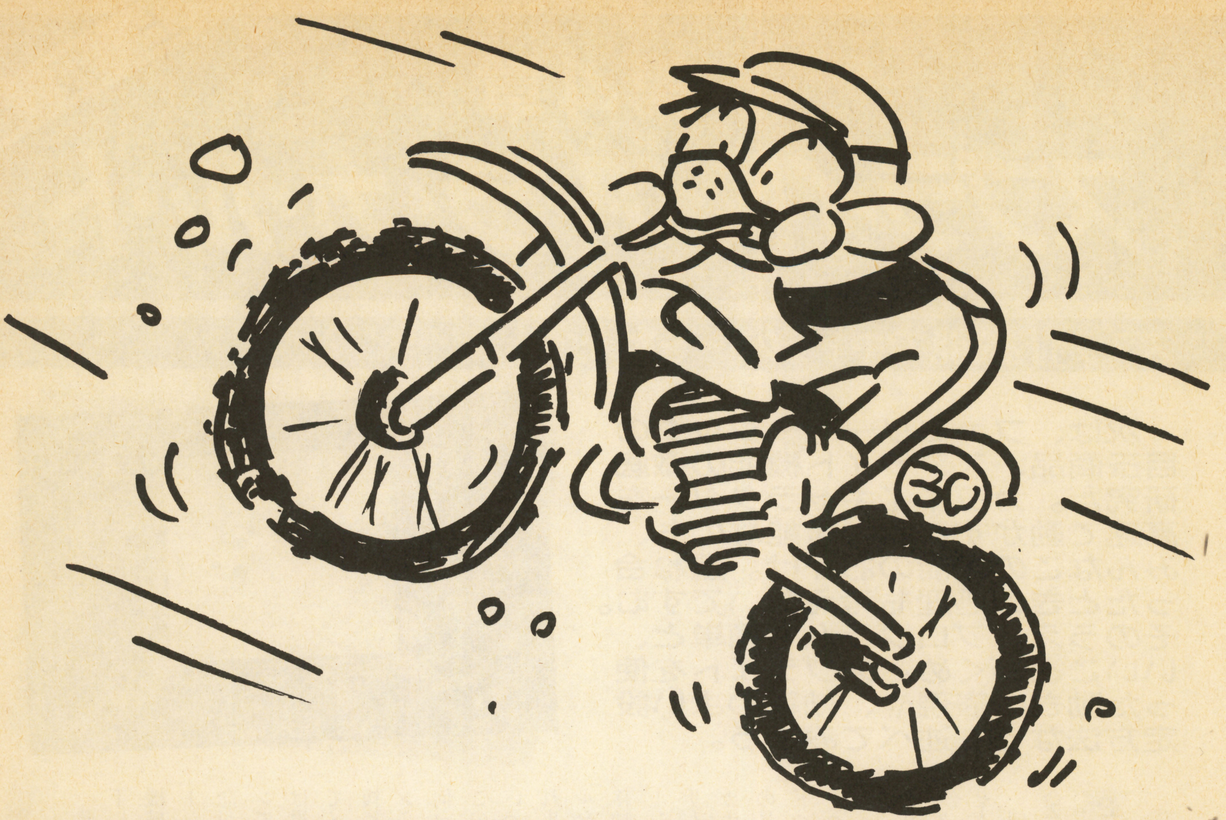
と、ここまで書けば結果は見えている。つまり、3色ぐらいの配色番号を使って、背景を動かしたい順に描いておく。そして、すべてを画面全体の色と同じにしておいて、1つずつ順に色をつけていく。そして1つ前を画面色にもどしていく。こうすれば大きな図形もかなり高速に動かすことができる。

(4)ゲーム入力法

ゲームリストを打ちこんでまずセーブ。次にBGグラフィックスで背景画面を打ちこむ。このとき、キャラクターのコードだけでなく配色番号にも気をつけてほしい。でないと背景が動かなくなってし

かっとびモトクロッサー

```
100 DIMD(&H2,&H2)
110 CGSET &H1,&H0
120 VIEW:CGEN&H3:SPRITEON
130 Z=&H1:X=116:Y=180:R=&H0
140 DEFSPRITEZ,(R,Z,R,R,R)=CHR$(222)+CHR$(223)
150 DEFSPRITER,(R,R,R,R,R)=CHR$(186)
160 DEFSPRITE&H2,(R,R,R,R,R)=CHR$(19)
170 DEFSPRITE&H3,(R,Z,R,R,R)=CHR$(200)+CHR$(201)
180 DEFSPRITE&H4,(R,Z,R,R,R)=CHR$(204)+CHR$(204)
190 DEFSPRITE&H5,(R,Z,R,R,R)=CHR$(195)+CHR$(195)
200 DEFSPRITE&H6,(&H3,R,R,R,R)=CHR$(238)
210 DEFSPRITE&H7,(&H3,R,R,R,R)=CHR$(238)
220 R=R+&H1:LOCATE13,&H1:PRINTR;:G=401:H=&H5:L=&H0
230 POKE&H4015,&H1:POKE&H4000,&H3F:POKE&H4002,&HFF,&H9:POKE&H4001,&H7F
240 PALETB Z,31,31,31,31
250 Z=Z+&H1:IFZ=&H4THENZ=&H1
260 PALETB Z,31,48,48,48
270 G=G-&H1:SPRITE&H7,191+(400-G)/10,31
280 V=&H0:A=STICK(V):IFA=&H1THENV=&H2
290 IFA=&H2THENV=-A
300 IFV=&H0THENH=H+&H1:IFH=65THENH=64
310 IFV THENH=H-&H1:IFH=&H4THENH=&H5
320 SPRITE&H6,24+H/&H2,31:POKE&H4002,(64-H)*4
330 SPRITE&H1,X,Y:SPRITE&H0,X+&H4-V,Y+&H8:SPRITE&H2,X+&H6+V,Y+&H4:X=X+&H2*V
340 IFX<48ORX>200THENH=&H5
350 IFY<180 THEN Y=Y+&H2
360 B=Z-&H1:IFD(B,&H0)=&H0THEN400
370 C=D(B,&H0):D=D(B,&H1):E=D(B,&H2):F=&H2+Z
380 C=C+E:D=D+D/&H8:IFD>220THENC=&H0:SPRITEF:GOTO430
390 GOTO420
400 IFRND(10)>R THEN450
410 C=RND(&H3)*&H8+116:D=100:E=RND(10)*&H2-&H9:F=&H2+Z
420 SPRITEF,C,D
430 D(B,&H0)=C:D(B,&H1)=D:D(B,&H2)=E
440 IFY=180ANDD=177ANDABS(C-X)<15THEN480
450 L=L+H:IFL>22000THEN550
460 IFG=&H0THEN570
470 GOTO240
480 ONZ GOTO490,530,510
490 X=X-&H8-(X>C)*16:Y=178
500 GOTO450
510 H=&H5:X=X+RND(&H3)*&H8-&H8:Y=176:GOTO450
520 GOTO450
530 H=H-&H10:IFH<&H5THENH=&H5
540 Y=170:GOTO450
550 POKE&H4001,&HFC,&HD4,&H8:PAUSE200:IFR<10THEN220
560 LOCATE &H4,&H4:PRINT"YOU ARE GOOD RIDER !!";:GOTO580
570 POKE&H4001,&HF6,&HD4,&H8:LOCATE&H5,&H4:PRINT"FUELTANK IS EMPTY !!";
580 LOCATE10,16:PRINT"GAME OVER";:LOCATE&H0,22:END
```



BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	S	P	E	E	D								R	A	N	K								F	U	E	L	
1																												
2	0			M	A	X																	M	A	X			0
3																												
4																												
5																												
6														K71	L01													
7													K72	L22	L12	L02												
8												K73	L22			L12	L03											
9											K73	L23					L13	L03										
10										K71	L23							L13	L01									
11									K71	L21									L11	L01								
12								K72	L21											L11	L02							
13							K72	L22													L12	L02						
14						K73	L22															L12	L03					
15					K73	L23																	L13	L03				
16				K73	L23																			L13	L03			
17			K71	L21																					L11	L01		
18		K71	L21																							L11	L01	
19	K71	L21																									L11	L01
20	L21																											L11

まう。背景もセーブしたら、RUN↲でスタートだ。

⑤遊び方

BEEP関係者の若い衆(気ばかりの人も若干いるが)の間で、最近バイクが大はやり。チーフは、いつも冷や汗ばかり。「右手だけは帰ってこいよ!!」これは、ツーリング前に山幸彦がチーフからいただいたアリガターイお言葉。確かに、危なっかしい乗り物だけど、メカメカしたマシンと一体となって風を切る感じはやはり魅力だ。

そんな感じをゲームの上で出したくて作ったのがこのゲーム。荒れ地をカッとばすモトクロスをイメージしたんだけど……。

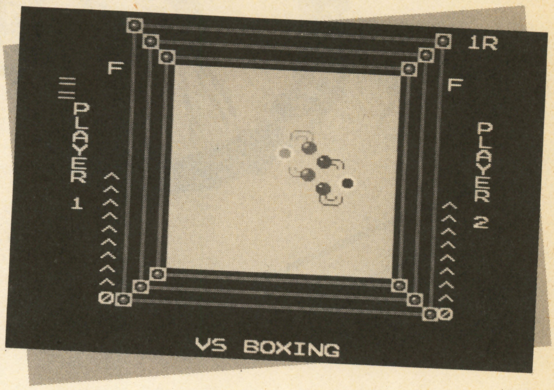
ルールは、燃料がつきるまでに、一定距離を走りぬけること。アクセルはなし。つねに開きっぱなしだ。直線で進めばグングン加速し、すぐに最高速だ。しかし、障害物がそれを許さない。✚ボタンの左右ですばやく回避しよう。障害のうち、レンガは踏んづけるとスピードを大幅に失う。四角のジャンプ台は、次の障害をとびこえられるが、スピードも少し失う。三角の突起は、スピードを殺さずコースを変えられるので利用しよう。さあ、コースを駆ける狼となれ。

上級編

第4章
スプライト・ア・ラ・カ・ル・ト

スプライト機能でキャラクターを動かす

次は、ファミリーベーシックの目玉商品、スプライト機能の徹底研究だ。カワイイキャラクターを高速で動かすことが可能なのも、みんなこの機能のおかげ。重ね合ったときの処理も考えないですむ。そのうえ、プログラムも簡単と、いいことづくめ。スプライトを使った動きの表現や、効率のよい設定方法などを述べてみよう。



(1) スプライトの利点

最初にファミリーベーシックを手にしたとき、何とも中途半端(ちゅうとはんぱ)な機械だなと思ったものである。ベーシックは遅いという確固たる信念が私にはあったので、このマシンで動きのあるゲームを作ろうとは思ってもいなかった。かといってメモリも2Kしかないのだから、複雑な計算や大きな地図の入ったシミュレーションなどもできない。何を作ろうかと考えていたところ、このスプライトという命令に目がいった。当時、MSXなどで使われ出した命令だ。ためしにサンプルを作ってみると、これが意外に移動が速い。キャラクターのパターンも固定されているので、メモリも少なくてすむ。こいつは、ナカナカいけるぞ!!と思ったのが運の尽き。ファミリーベーシックにドップリとつかる運命となってしまった。

一度、設定してしまえば、座標を与えるだけでドット単位でキャラクターを動かせるし、重ね合わせも優先度にそってやってくれる。何と高速、省メモリ、お手軽な命令だろう。私は大好きである。

(2) スプライトの設定法

DEFSPRITE。これがスプライトを設定する命令である。これで設定するときには、スプライト番号、配色番号、キャラクターの大きさ、表示優先度、X・Y軸の反転指示などを指定した後、指定したいキャラクターのコードを文字列の形で代入する。この処理は、はじめにやるだけでよいのだが、結構メモリを使ってしまうことが多い。そこで処理をできるだけ短くする方法をいくつか考えた。

ファミリーベーシックでは、数字は、値にかかわらず必ず3文字分のメモリを消費する。ところがDEFSPRITE命令には0と1の値が山のように出てくる。そこで、AやBなどの変数に0、1の値を入れて、数字の代わりにすれば、変数はその文字数分しかメモリをとらないので効果的だ。

その他には、文字のコードをDATA文にして次々とREAD文で読みとっていくなどいろいろあるが、極めつけは、次の項で述べるPOKE文による方法だろう。

(3) 動きを表現する方法

スプライト1枚で動くキャラクター(スターシップやスターキラー)は、ただ位置をSPRITE文で指定してやるだけでよい。しかし、マリオやペンペンが、氷の上をすべるようにスーッと移動したらブキミだ。やはり手をふり足を動かして動かなくては。それには、いくつかの動きのパターンを次々と表示してやる必要がある。

まずオーソドックスな方法として、複数のスプライトを利用する方法がある。2つ以上のスプライトにそれぞれ違ったポーズのキャラクターを定義しておき、1枚を表示。そして消す。次に少し移動させて、次のパターンのスプライトを表示させる。その繰り返しを行えば、アニメーションで手足を動かしながら、移動しているように見えるわけである。これは、スプライトの定義をプログラムの最初に行ってしまうので、かなりスピーディに行うことができる。しかし、スプライトを2枚以上使ってしまうので、多くのものを動かしたいときには、スプライトが足りなくなってしまう。

次に、スプライトを1枚でやる方法を考えてみよう。まずは、キャラクターが移動するたびに、DEFSPRITE文でキャラクターのポーズを変え、SPRITE文で移動させる方法だ。この方法ならキャラクターのデータテーブルをDATA文や文字列の中に入れておいて、動くたびにポーズを変えるので、1枚のスプライトで

できてしまう。しかしこの方法は、試してみるとわかるのだが、相当スピードが遅くなるうえに、時々画面がチラついてしまう。あまりいい方法とはいえない。

そこで次に考えたのが、スプライトのワークエリアを直接書き換えてやろうというもの。これは、ファミリーベーシックの内容をセッセと逆アセンブルして解読したときに気がついた方法だ。

DEFSPRITE文で、スプライトをセットしたとき、そのキャラクターは、&H680番地のメモリから4文字ごとに収められるということがわかった。そこで、この内容を直接変えてやれば、表示が変わるのではないかと考えたわけだ。この方法は、DEFSPRITEで最初のキャラクターを設定するときにも利用できる。DEFSPRITE0、(0,1,0,0,0)=〝〟のように設定しておいて、あとでワークエリアに直接書きこむのだ。

このやり方は、まずスプライト番号からワークエリアを計算することから始まる。スプライト番号をN(0~7)とすると、ワークエリアは、&H680+4*Nの番地から4バイトということになる。

次に、キャラクターのコードをメモリに書きこむわけだが、それには、POKEという命令が役に立つ。第6章で詳しく説明するが、メモリのアドレスを指定して、内容を変更する命令である。しかも、内容を連続して与えれば、その次のアドレスの内容も変えてくれる。

例えば、

POKE&H684,0,1,2,3,

とすれば、スプライト番号1のワークエリアに、マリオの歩いているパターンを書きこめる。

このPOKE文とSPRITE文を次々と行うことで、すばやくパターンを変更することができる。

(4) 入力法

メモリいっぱいにつめこんだプログラムなので見にくくなっているが、そのとおり打ちこんでセーブしてほしい。プログラムは、POKE文でメモリを直接操作しているので、暴走するとすべてオジ

VS2次元ボクシング リスト1

```
100 Z=&H0:V=Z:Y=&H1:VIEW:LOCATE&H9,23:PRINT"VS BOXING";:CGEN&H3:CGSETY,Y:PALETSY
,20,32,20,23:SPRITEON:FORA=Z TO&H7:DEFSPRITEA,(Y+(A>&H3),Y,Z,Z,Z)="":NEXT:FORA=&
H680TO&H69F:READB:POKEA,B:NEXT:O=Z:D=O:L=O:R=16:S=R
110 V=V+Y:LOCATE24,Z:PRINTV"R";:T=599:GOSUB420:PAUSE100:PLAY"YOM1V1503C7:YOM1V15
05C7":POKE&H4015,&H7:POKE&H4000,15,&H84:POKE&H4004,15,&H99
120 T=T-Y:IF(T/100)*100=T THENLOCATEZ,&H5-T/100:PRINT" ";:IFT=Z THEN430
130 O=Y-O:SWAPA,I:SWAPB,J:SWAPC,K:SWAPD,L:IFR*S=Z THEN370:DATA32,222,9,17,12,24,
32,224,207,32,32,32,207,32,225,32,32,207,32,32,32,207,32,224,223,32,9,17,12,24,2
25,32
140 GOSUB360:M=STICK(O):P=Z:Q=P:IFM ANDY THENP=&H8:DATA223,207,32,207
150 IFM ANDY THENP=&H8:DATA223,207,32,207
160 IFM AND&H2THENP=-&H8:DATA207,32,207,225
170 IFM AND&H4THENQ=&H8:DATA222,207,32,207
180 IFM AND&H8THENQ=-&H8:DATA207,32,207,224
190 IFD THENP=D:D=Z:DATA207,223,9,253,12,253,207,225,32,32,227,242,227,240,32,32
200 A=A+P:B=B+Q:IFB<47ORB>135THENB=B-Q:DATA32,32,241,227,239,227,32,32,222,207,2
53,17,253,24,224,207
210 IF(O=Z)ANDA>160OR(O ANDA<64)THENA=A-P
220 IFA=48ORA=176THEND=&H8-16*O
230 H=&H688+O*&H9:M=STRIG(O)
240 IF C=Z ANDM AND&H4THENON(PEEK(H)=207)+&H2+O*&H2RESTORE140,160,170,180:FORE=H
 TOH+&H6STEP&H2:READF:POKEE,F:NEXT:C=&H8:GOTO260
250 C=C/&H2
260 IF(M AND&H8)=&H0THEN120
270 IF(A-I)*SGN(O*&H2-Y)<>&H8ORABS(B-J)>24THEN310
280 E=(B-J+24+(PEEK(H)=207)*24)/&H8:IFE<>Y ANDE<>&H2THEN310
290 IFPEEK(&H691-O*&H9+&H2*E)=207THEN330
300 L=&H8-16*O:U=Y
310 POKE&H4006,&H33,&H9:E=H+&H6+(PEEK(H)=207)*&H6-O:IFO THENPOKEE,207,227:GOSUB3
60:POKEE,32,207:GOTO330
320 POKEE,227,207:GOSUB360:POKEE,207,32
330 IFU=Z THEN120
340 POKE&H4002,&H92,&H9:U=Z:IFO THENR=R-Y:LOCATE&H3,19-R:PRINT" ";:GOTO120
350 S=S-Y:LOCATE24,19-S:PRINT" ";:GOTO120
360 FORG=O*&H4TOO*&H4+&H3:SPRITEG,A+((G AND&H3)/&H2)*16,B+(G ANDY)*16:NEXT:RETUR
N
370 IFR=Z THENH=&H680:RESTORE190:GOTO390
380 H=&H690:RESTORE200
390 E=A:F=O*&H2-Y:FORA=E TOE+F*24STEP&H4*F:GOSUB360:NEXT:PLAY"T403C2":FORE=H TOH
+15:READF:POKEE,F:NEXT:GOSUB360:SWAPA,I:SWAPB,J:O=Y-O:GOSUB360:PLAY"D":LOCATE&H8
,22:PRINT"PUSH START!"
400 IFSTRIG(Z)=Y THENRUN
410 GOTO400
420 A=56:B=135:I=168:J=47:GOSUB360:O=Y-O:SWAPA,I:SWAPB,J:GOSUB360:O=Y-O:SWAPA,I:
SWAPB,J:RETURN
430 PLAY"YOM1V1503C7:YOM1V1505C7":GOSUB420:FORA=Z TOY:IFR<16THENR=R+Y:LOCATE&H3,
20-R:PRINT"^";
440 IFS<16THENS=S+Y:LOCATE24,20-S:PRINT"^";
450 FORB=Z TO&H2:LOCATEZ,B*&H2+A:PRINTCHR$(227);:NEXT:BEEP:NEXT:GOTO110
```

ボクシング　(おまけのバージョンV3用拡張リスト)

```
10 CLS:LOCATE&H5,&H5:PRINT"VS BOXING    V2.0":LOCATE&H8,11:PRINT"ONE PLAYER":LOCA
TE&H8,12:PRINT"TWO PLAYERS":LOCATE20,20:PRINT"BY KAZ"
20 X=1
30 LOCATE&H5,12-X:PRINT"->";:A=STRIG(0):IFA=2THENLOCATE5,12-X:PAUSE20:PRINT"  ";
:X=1-X
40 IFA<>1THEN30
145 IFO*X THEN500
235 IFO*X THEN690
500 IFR<&H4 AND S>&H7THEN640
510 E=(J-B+24+(PEEK(H)=207)*24)/&H8
520 IFS<10 THEN590
530 IFT>299 THEN 640
540 ONRND(&H4)+&H1GOTO550,560,570,580
550 M=&H2:GOTO150
560 M=&H4:GOTO150
570 M=&H8:GOTO150
580 M=Y:GOTO150
590 IFABS(J-B)>25THEN540
600 IFA-I>32THEN540
610 M=Y
620 M=M OR&H8:IFE<&H2THENM=(M OR&H4)AND&H7
630 GOTO150
640 IFRND(&H5)>&H2THEN540
650 E=(B-J+24+(PEEK(&H691)=207)*24)/&H8
660 M=&H8:IFE<&H2THENM=&H4
670 M=M OR&H1:IFA-I>8THENM=(M OR&H2)AND&HE
680 GOTO150
690 M=&H8:IFRND(5)<3THENM=&HF
700 IFABS(A-I)>24ORABS(B-J)>24THENM=M AND&H7
710 GOTO240
```

V3を持っている人は，このプログラムを打ちこめば，コンピュータ相手の1人プレイもできます。
テープに入っているプログラムをロードしたあと，このプログラムを打ちこんでください。
正常に動くことを確認したら，別のテープにセーブしましょう。BG面も同時にセーブしておけば，以降このテープからプログラムとBG面をロードするだけで，ゲームが楽しめます。

1人，2人プレイの選択方法

RUNさせるとメニューがでるので，セレクトキーで1人プレイか2人プレイを選んで，スタートボタンを押せば，ゲームが始まります。

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	J30				D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40				
1	J30				K60	D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40	K60				
2	J30				K60	K60	D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40	K60	K60				
3	J30			F	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	F			
4	J30			∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
5	J30			∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
6		P		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		P	
7		L		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		L	
8		A		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		A	
9		Y		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		Y	
10		E		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		E	
11		R		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		R	
12				∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
13		I		∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧		2	
14				∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
15				∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
16				∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
17				∧	K60	K60	K60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	K60	K60	K60	∧			
18				∧	K60	K60	D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40	K60	K60	∧			
19				∧	K60	D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40	K60	∧			
20				Ø	D40	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	K50	D40	Ø			

ヤンになる可能性がある。背景も入力しセーブして、RUN⏎だ。

(5)遊び方

このプログラムは、背景、スプライトともにテーブルBのキャラクターを利用している。リングを上から見た2人用ボクシングゲームだ。

プレイヤー1は左下コーナーから、プレイヤー2は右上コーナーからファイト開始。移動は✥ボタンで上下左右ナナメに動ける。Aボタンでパンチを出して、Bボタンでガードとパンチの腕を左右交代する。頭の前方に曲げた手がガードだ。

出したパンチが相手の顔面をとらえるとヒット!! 相手の体力が1つ減り、相手は後ろによろめきさがる。体力はスタート時に16の∧で画面左右に示されていて、ダメージをうけると減っていき、0になるとKOだ。

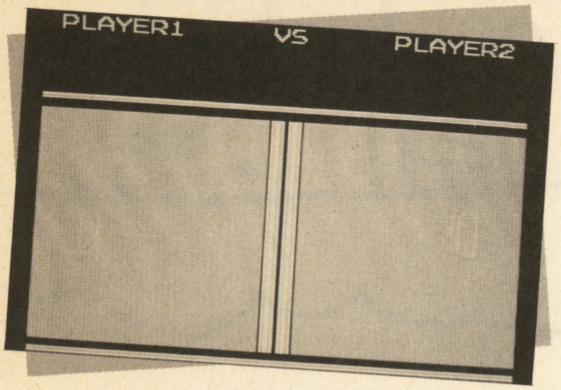
また、画面左上の6つの"="はラウンドの残り時間を示していて、なくなると1ラウンド終了。体力が2だけ回復する。

ロープの反動やフットワーク、左右の手のガード交換で、華麗に舞い、相手にダメージを与えよう。目ざせ、世界チャンプを!!

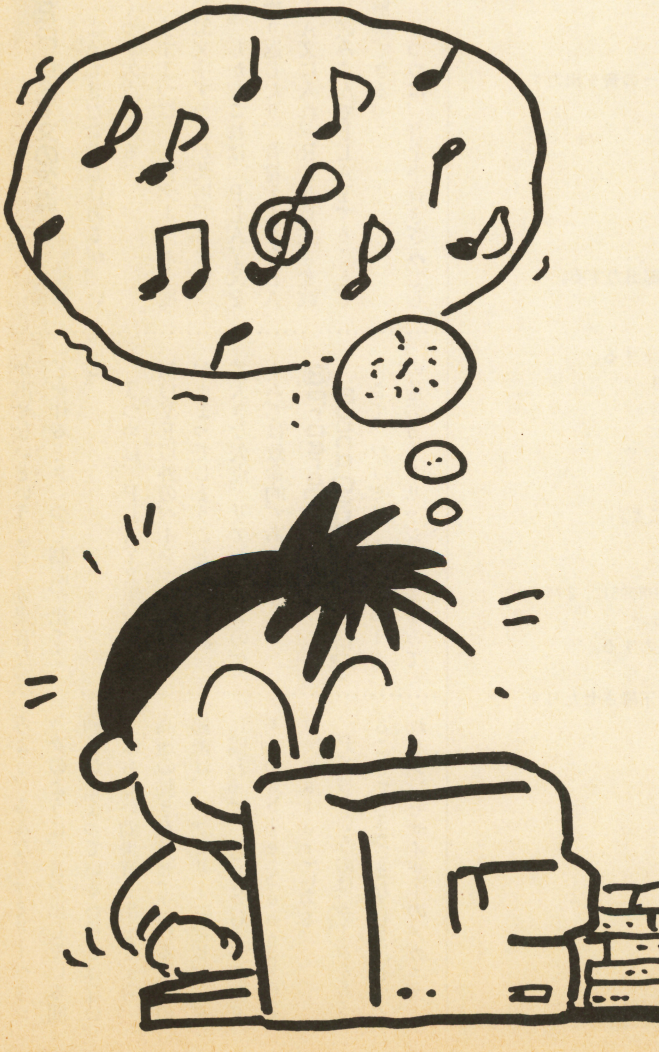
上級編

第5章 PSGをコントロール

PLAY文で音階フル活用



きれいな画面と楽しい音楽。これがゲームに必要な調味料。
この章では、ファミリーベーシックの音楽機能について、取り上げてみた。ＰＬＡＹ文での音の出し方、ＰＬＡＹ文の欠点、ファミリーコンピュータ本体にあるＰＳＧの説明、さらに、ベーシックからのＰＳＧの使用法と話を進めていこう。



(1)PLAY文使用法

ちゃんとした音階で曲を演奏するための機能は、もうファミリーベーシックに用意されている。ＰＬＡＹ文がそれだ。その後ろに文字列または、文字変数で音符データを指示すれば、その通り演奏してくれる。楽譜を見ながらでも簡単にデータ化できる。

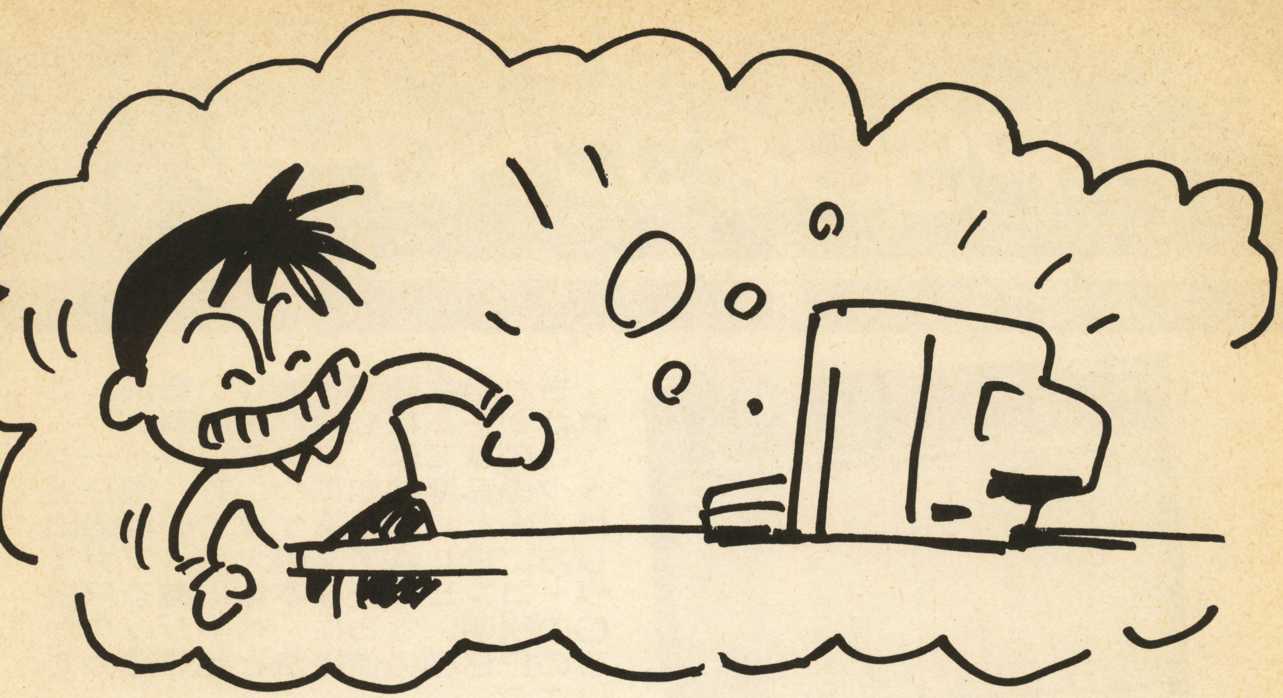
音階のほかにも、いろいろな機能がついている。まず、テンポの指定。曲全体の演奏する速さを変えよう。Ｔ1（速い）～Ｔ8（遅い）を指定する。ＭＯＶＥ文と同時に動かすときは、曲のスピードが半分になるのでテンポを速くセットすることが必要だ。

次は音色の指定Ｙ0～Ｙ3で、いろいろと音色を変えることができる。曲のイメージにあった音色で演奏が可能だ。

エンベロープは音の余韻(よいん)が消えていくスピードをセットする機能。Ｍ1と指定して、Ｖ0～Ｖ15で音の長さを決める。数字の大きいほうが長い音が出る。これにより、スタッカートなどの効果も表現できる。Ｍ0とすると、Ｖ0～Ｖ15で音の大きさを指定することになる。

音階はＯ0（低い）～Ｏ5（高い）の6オクターブで指定し、音符を決めてその後に0（32分音符）～9（全音符）で長さを決めることができる。休符はＲである。

以上の機能を十分生かせれば、それらしい音を出せるわけだ。



(2)PLAY文の欠点

よくできた命令ではあるが、いくつか欠点もある。

まず1つは、音符データが1度に31文字ずつしか使用できないこと。しかしこれは、PLAY文を繰り返したり、音符データをDATA文に入れ、READ文で変数に読みこんでPLAYすることで解決できる。

2つめは、音を半音きざみでしか出せないということ。効果音などを作るとき、少し制限が出てくる。

3つめは、PLAY文実行中に、MOVE以外のベーシックの動作が止まってしまうこと。これが一番大きな欠点。リアルタイムのゲームでは致命的である。

(3)Programable Sound Generatorとは?

さて、ファミリーコンピュータの他のカートリッジゲームをみてみよう。みんなファミリーベーシックのように、音を出している間動きが止まっているわけではない。すばらしい効果音を出しながら、高速の動きを見せてくれる。その秘密は、ファミリーコンピュータに備わっているPSG(Programable Sound Generator)にある。

PSGというのは、データを与えることでさまざまな音を出してくれるシンセサイザーのメカニズム

PSG コントロール法

①ミキサーコントロール（アドレス& H4Ø15)

・チャンネルA、B、CのVCO、ノイズジェネレーターの音を出力するかどうかを決定する。

・bit	7	6	5	4	3	2	1	Ø
内容	Ø	Ø	Ø	X	Ø	X	X	X
				ノイズ		Ch.C	Ch.B	Ch.A

X=Øでoff、X=1でOn（オン）

・例）POKE & H4Ø15、&H17 ChA、B、Cとノイズを全部出力する。

②エンベロープ、ボリューム、音色コントロール。
（アドレス & H4ØØØ+4＊n)

・PLAY文のデータのY、M、Vに相当する機能をセットする。

・bit		7	6		5	4	3	2	1	Ø
内容	YØ	Ø	Ø	M1	Ø	Ø	X	X	X	X
	Y1	Ø	1							
	Y2	1	Ø	MØ	1	1				
	Y3	1	1							
							VØ～15			

・例）POKE & H4ØØØ、SHØF PLAY"YØM1V15"と同じこと。

③周波数セット （アドレス&H4ØØ2+4＊n…下位 / &H4ØØ3+4＊n…上位）

・音の高低をセットする。テーブルは、&H8ØØ6～&H8Ø95に、2バイトずつO Ø C～O 5 Bまで半音きざみに書かれている。

・例）POKE &H4ØØ2、&HD4、&HØ8 chAに、O 3 Cを出力する。

④効果音のコントロール（アドレス &H4ØØ1+4＊n)

・音を単音にするか、音の周波数を上昇させたり、下降させたりするかを決める。

・bit	7	6	5	4	3	2	1	Ø
単音	Ø	X	X	X	Ø	X	X	X
シフト	1				1			
		シフトする速度Ø～7、大ほどゆっくりシフトする。			ØUP 1DOWN	シフトするステップ1～7、大ほど細かくシフトする。		

・例）POKE &H4ØØ1、&H7F ChAを単音にセット、
POKE &H4ØØ5、&HBF ChBをシフトUPする。

㊟ 表のnは、Ø、1、2でChA、B、Cをコントロールしている。

のことで、最近のコンピュータには、ほとんどこれがつけられている。構造としては、まず音の源を発生するVCO(Voltage Controlled Oscillator)。与えられたデータによって周波数の変わる音を発信する発振器である。

そして、音色を決めるフィルターVCF(Voltage Controlled Filter)。音にかかるフィルターを変化させ音色を設定する。

さらに音の出方、消え方を決めるVCA(Voltage Controlled Amplifier)。データによって音の大小を決める増幅器。

これらの3つの機能を持ち、それぞれがコンピュータから簡単なデータを送るだけでコントロールできるようになっている。

さらに、さまざまな音の変化のバリエーションをもたせるため、エンベロープジェネレーター(音量などを変化させるパターンを発生する)やノイズを発生するノイズジェネレーター、複数のオーディオ信号を混ぜ合わせて出力するオーディオ・ミキサーなどの機能が備わっているものも多い。

このPSGをうまくコントロールすることさえできれば、すばらしい効果音が手に入るわけだ。

⑷ファミリーコンピュータのPSGの利用法

ファミリーコンピュータのPSGについては、詳しいことが発表されているわけではないので、その内容は定かでないが、かなりの機能を備えたすばらしいものらしい。ベーシックのPLAY文を解析したものに、2、3実験を加え、最低限の使い方だけ解明してみた。

①ミキサーの設定…&H4015に書き込む内容で、チャンネルA、B、Cの音とノイズジェネレーターの音を出すかどうかを決定する。ビット0、1、2、4がそれぞれチャンネルA、B、C、ノイズに対応していて、0でOFF、1でONである。

これ以後は、n＝0、1、2でチャンネルA、B、Cをそれぞれコントロールできる。

②エンベロープ、ボリューム、音色の設定…&H4000+4＊nのビット7と6でPLAY文のYにあたる音色を、ビット5、4が0ならM1、1ならM0をセットしビット0~3でV0~15をセットする。

③周波数セット…&H4002+4＊n(下位)&H4003+4＊n(上位)に、&8006~&H8095までのデータを書き込むとO0C~O5Bまで半音きざみに音がでる。

④効果音設定…&H4001+4＊nのビット7を0にすると単音、1にすると周波数の上昇、下降する音を出力できる

表に各内容を略記してみた。

⑸ゲーム入力法

まずは、ベーシックを何もない状態でスタートさせて、リストを打ちこみ、必ずセーブ。POKE文を使ったプログラムは、ワークエリアや自分を破壊する可能性があるので特に注意が必要だ。BGグラフィックスも入力してセーブしておこう。

⑹遊び方

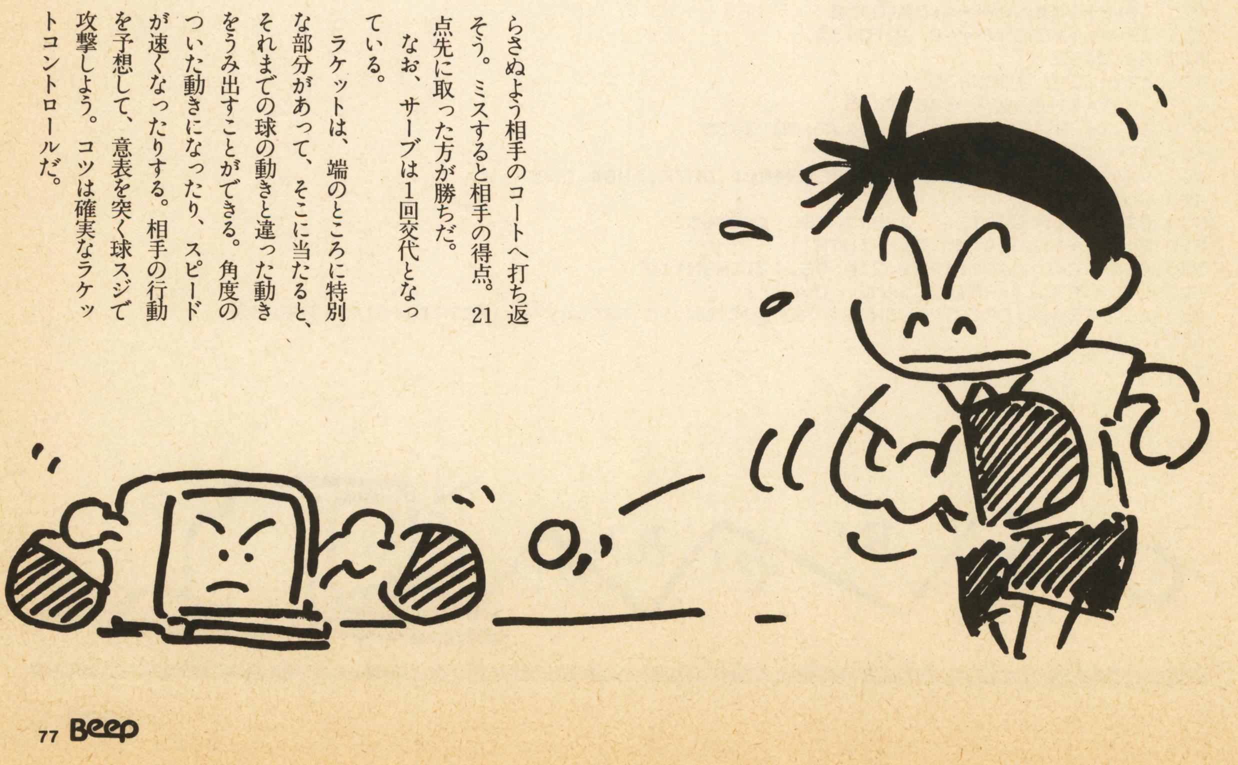
このゲームは、むかしなつかしい2人用テレビピンポンである。いろんな音を出しながら飛びまわる球をラケットで打ち返して、相手のミスを誘うゲームだ。

RUN↵でゲームスタート。プレイヤー1が左側、プレイヤー2が右側のラケットを操作する。ラケットの操作は✜ボタンの上下で行える。

ゲームスタートすると、まず、サーブから。ボールの現れたほうのプレイヤーがAまたはBボタンでOK。あとはボールを後ろにそらさぬよう相手のコートへ打ち返そう。ミスすると相手の得点。21点先に取った方が勝ちだ。

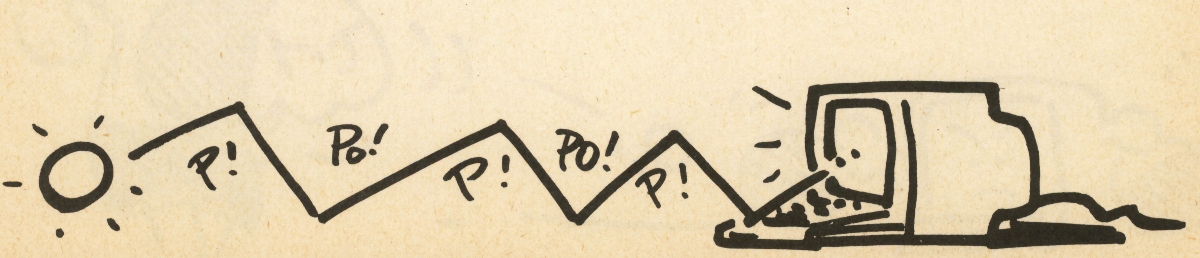
なお、サーブは1回交代となっている。

ラケットは、端のところに特別な部分があって、そこに当たると、それまでの球の動きと違った動きをうみ出すことができる。角度のついた動きになったり、スピードが速くなったりする。相手の行動を予想して、意表を突く球スジで攻撃しよう。コツは確実なラケットコントロールだ。



むかしなつかしVSテレビピンポン

```
10 CGEN3:CGSETO,0:PALETB 0,47,48,33,2:PALETS 0,47,48,40,56:PLAY"T1:T1":VIEW
20 LOCATEO,21:FORI=0TO55:PRINTCHR$(254+66*(I>27));:NEXT
90 S1=0:S2=0:SB=1:X1=32:X2=208:Y1=120:Y2=120
110 BX=124-100*SB+50:BY=132:VX=SB*8-4:VY=RND(2)*8-4:SB=ABS(SB-1):I=1:SPRITE ON:A
$=CHR$(239)+CHR$(240)+CHR$(238)+CHR$(238):B$=CHR$(238)+CHR$(238)+CHR$(241)+CHR$(
242):DEFSPRITEO,(0,0,0,0,0)=CHR$(207)
150 DEFSPRITE1,(0,1,0,0,0)=A$:DEFSPRITE2,(0,1,0,0,0)=B$:DEFSPRITE3,(0,1,0,0,0)=A
$:DEFSPRITE4,(0,1,0,0,0)=B$
170 SPRITEO,BX,BY:SPRITE1,X1,Y1:SPRITE2,X1,Y1+16:SPRITE3,X2,Y2:SPRITE4,X2,Y2+16
180 IFI=1 AND STRIG(SB)=0 THEN180
190 IF I=1 THEN POKE&H4015,&H17:POKE&H4004,&HF6,(VY<&H0)AND&H80R&HF5,&HD4,&H8:PO
KE&H4000,&HF,&H7F
196 I=0:X=BX+VX:Y=BY+VY:G=SGN(VX)*4:IFABS(VX)=12THENG=0
210 IFY<64ORY>196THENVY=-VY:Y=BY+VY:POKE&H4002,&HFF,&H9,&HF6,((VY<0)AND&H8)OR&HF
5
220 IFX<16THEN500
230 IFX>232THEN510
240 IFX>28ANDX<44THEN400
250 IFX>204ANDX<220THEN420
260 BX=X:BY=Y:A=STICK(0)AND12:B=STICK(1)AND12:IFA=4THENY1=Y1+4
290 IFA=8THENY1=Y1-4
300 IFB=4THENY2=Y2+4
310 IFB=8THENY2=Y2-4
320 IFY1<64THENY1=64
330 IFY1>172THENY1=172
340 IFY2<64THENY2=64
350 IFY2>172THENY2=172
360 GOTO170
400 IF(Y<Y1)OR(Y>Y1+24)THEN260
403 IFY>=Y1+4THEN408
404 IFVY=8THENVY=4:VX=VX+G:GOTO419
405 IFVY=4THENVY=-4:GOTO419
406 IFVY=-4THENVY=-8:GOTO419
407 GOTO419
408 IFY<Y1+20THEN419
409 IFVY=4THENVY=8:GOTO419
410 IFVY=-8THENVY=-4:VX=VX+G:GOTO419
411 IFVY=-4THENVY=4:GOTO419
419 VX=ABS(VX):X=BX+VX:POKE&H4001,&H7F,&HD4,&H8:GOTO450
420 IF(Y<Y2)OR(Y>Y2+24)THEN260
421 IFY>=Y2+4THEN426
422 IFVY=8THENVY=4:VX=VX*2:GOTO435
423 IFVY=4THENVY=-4:GOTO435
424 IFVY=-4THENVY=-8:GOTO435
425 GOTO435
426 IFY<Y2+20THEN435
427 IFVY=4THENVY=8:GOTO435
428 IFVY=-8THENVY=-4:VX=VX*2:GOTO435
429 IFVY=-4THENVY=4:GOTO435
435 VX=-ABS(VX):X=BX+VX:POKE&H4001,&H7F,&HD4,&H8
450 GOTO260
500 S2=S2+1:LOCATE25,2:PRINTS2;:GOTO520
510 S1=S1+1:LOCATE5,2:PRINTS1;
520 PLAY"C4D:C4A":IFS1<21ANDS2<21THEN110
525 LOCATE9,13:PRINT"GAME  OVER";
530 LOCATEO,4:PRINT"PUSH START BOTTUN TO REPLAY";:IFSTRIG(0)<>1THEN530
550 RUN
```



BGグラフィック画面データ

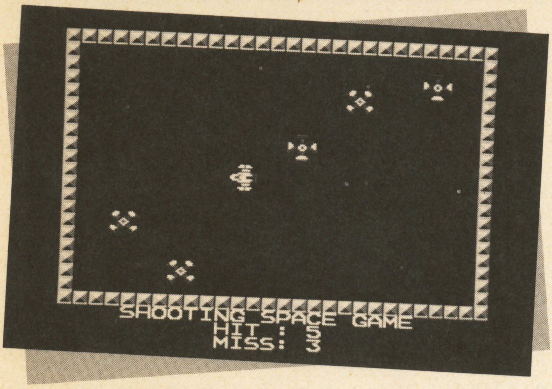
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0		P	L	A	Y	E	R	I						V	S						P	L	A	Y	E	R	2	
1																												
2																												
3																												
4																												
5	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50	E50
6	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
7	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
8	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
9	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
10	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
11	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
12	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
13	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
14	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
15	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
16	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
17	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
18	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
19	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60
20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	E20	E20	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60	M60



第6章 機械語を知ろう!

上級編

機械語はよりコンピュータに近い言葉



さてさて、上級編もそろそろ終わり、この本のオーラスとなってしまったわけだ。ここでは、思いきってベーシックを捨て、機械語の世界に足をふみこんでみた。全容を説明するには、この本ぐらいではまったく足りない広大な世界だ。とりあえず、概念だけ説明して、ベーシックとの関係について述べてみよう。

(1) 機械語って何かな

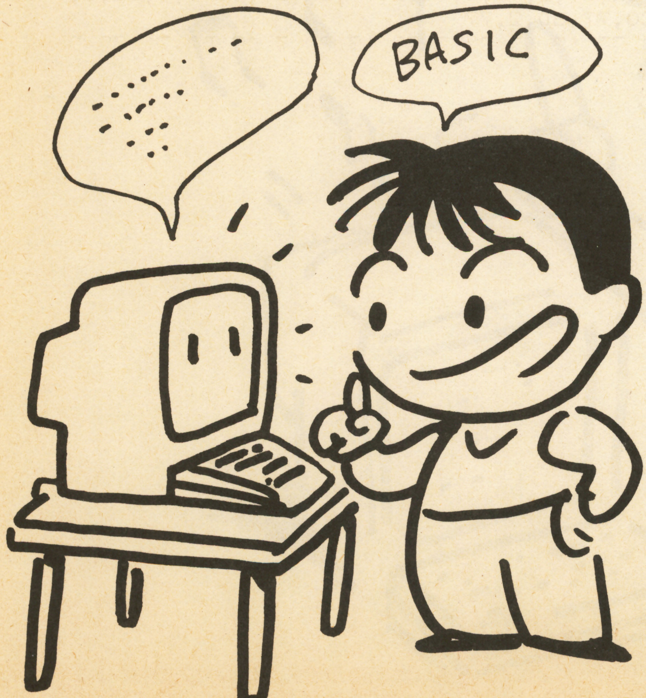
ファミリーベーシックで作ったプログラムはいくらうまく作っても、スピードという点では、カートリッジのゲームにはとうていかなわない。どうしてなのだろうか? そのあたりを、まずベーシックをもう一度ふりかえって考えてみよう。

一般にベーシックというのは、スピードはあまり速くないが、より人間の話す言葉に近い高級言語であるとされている。この高級という言葉は、プログラムしやすい、初心者に理解しやすいという意味あいをもっている。

では、それに対して低級言語というのがあるのだろうか? それにあたるのが機械語なのである。プログラムを組むということは、人間が、コンピュータにしたいことを理解させるということである。人間は、人間の言葉を使い、考え、コンピュータは機械語で仕事を行う。その間をつなぐのがプログラム言語なのである。コンピュータ側から、より人間に近づいていく順に、機械語、アセンブラ、ベーシックとより人間に理解しやすい形になっていく。しかし、人間に理解しやすくなっていくと、こんどは機械側からは、どんどん離れていってしまうことになる。ベーシックなどでは、その距離をコンピュータ自体にうめさせているのである。つまり、入力したプログラムをコンピュータが理解できるよう解析するプログラム、それがベーシック言語なのである。

しかしそのかわり、ベーシックの1つの命令を理解するために、コンピュータは多くの機械語の処理を必要とする。そこで、ベーシックのプログラムは遅いとなるわけである。

それに対して、カートリッジでは、プログラムそのものが機械語で書かれている。よって、すばらしい高速の動きが見られるのだ。



(2) 機械語の利点・欠点

ファミリーコンピュータを動かしているCPU(中央演算装置)は、すべて数字で仕事を行っている。高速の処理を、必要とするならば、人間のほうがコンピュータに近寄ってやる必要がある。プログラマーには、0~255(16進で00~FF)までの256個の数値の組み合わせでプログラムを作っていくわけである。そのため、プログラムの構造や動きを理解するには、かなりの努力と訓練が必要となる。そして、そのプログラムを組むためにかかる時間と手間は、ベーシックの数十倍にもなってし

高速シューティングゲーム　リスト1

```
10 CLEAR &H75FF:A=&H7600
20 READA$:IFA$<>"@"THENPOKEA,VAL("&H"+A$):PRINTHEX$(A);":";A$:A=A+1:GOTO20
25 READ A$:IFA$<>"@"THENA=VAL("&H"+A$):GOTO20
40 DATA 4C,C0,77,A9,0,85,28,20,7A,A6,A5,90,29,F0,8D,58,77,20,7A,A6,A5,90,29,F,D,
58,77,8D,58,77,60,0
50 DATA B0,5,49,FF,18,69,1,60,0,0,0,0,0,0,0,0
60 DATA AD,52,77,38,ED,54,77,20,20,76,8D,56,77,C9,C,90,2,18,60,AD,53,77,38,ED,55
,77,20,20,76,C9,C,90,2,18,60,CD,56,77,B0,4,A9,0,F0,2,A9,1,38,60
70 DATA BD,C,77,49,FF,18,69,1,9D,C,77,18,7D,A,77,4C,75,77,BD,D,77,49,FF,18,69,01
,9D,D,77,18,7D,B,77,4C,92,77,0,0,0,0
80 DATA 8D,51,77,A,A,A,8D,50,77,AA,DE,9,77,F0,1,60,BD,8,77,D0,3,4C,BA,77,C9,1,D0
,3,4C,C7,76,8A,4A,AA,A0,4,A9,90,9D,80,06,18,69,1,E8,88,D0,F6,A9,3,AE,50,77,38,FD
,8,77,9D,8,77,4C,D1,76
90 DATA 8A,4A,AA,A0,4,A9,94,4C,AE,76,BD,E,77,9D,9,77,4C,60,77,AD,02,77,49,FF,18,
69,01,8D,02,77,18,6D,00,77,4C,50,75,AD,03,77,49,FF,18,69,01,8D,03,77,18,6D,01,77
,4C,6D,75
100 DATA @,7760,BD,A,77,18,7D,C,77,C9,18,B0,3,4C,60,76,C9,D9,90,3,4C,60,76,9D,A,
77,85,19,8D,52,77,BD,B,77,18,7D,D,77,C9,1F,B0,3,4C,72,76,C9,A8,90,3,4C,72,76
110 DATA9D,B,77,8D,53,77,85,1B,AD,0,77,8D,54,77,AD,1,77,8D,55,77,20,30,76,90,3,4
C,B4,77,AD,51,77,4C,14,A8,A9,1,8D,5A,77,60,A9,2,D0,F8,60,60
120 DATA 10,07,49,FF,18,69,1,A2,1,C9,4,90,2,A9,4,CA,D0,5,49,FF,18,69,1,60,AD,52,
77,18,69,4,38,ED,4,77,20,20,76,C9,8,90,2,18,60,AD,53,77,18,69,4,38,ED,5,77,20,20
,76,C9,5,90,2,18,60,38,60
999 DATA @,@
1000 PRINT"SPACE2 ｦ LOAD ｼﾃｸﾀﾞｻｲ":END
```

リスト2

```
10 CLEAR &H747F:A=&H7500
20 READA$:IFA$<>"@"THENPOKEA,VAL("&H"+A$):PRINTHEX$(A);":";A$:A=A+1:GOTO20
25 READ A$:IFA$<>"@"THENA=VAL("&H"+A$):GOTO20
30 DATA20,3,76,29,F,AA,29,1,F0,3,EE,2,77,8A,29,2,F0,3,CE,2,77,8A,29,4,F0,3,EE,3,
77,8A,29,8,F0,3,CE,3,77,A2,0,AD,2,77,20,C0,77,8D,2,77
40 DATAA2,0,AD,3,77,20,C0,77,8D,3,77,AD,0,77,18,6D,2,77,C9,18,B0,3,4C,DA,76,C9,D
9,90,3,4C,DA,76,8D,0,77,85,19,8D,52,77,AD,1,77,18,6D,3,77,C9,1F,B0,3,4C,EC,76,C9
,A8,90,03,4C,EC,76
50 DATA8D,1,77,8D,53,77,85,1B,A9,8,8D,59,77,AE,59,77,BD,A,77,8D,54,77,BD,B,77,8D
,55,77,20,30,76,90,3,4C,B4,77,A9,8,18,6D,59,77,8D,59,77,C9,30,F0,3,4C,7A,75,A9,0
0,4C,14,A8
60 DATAA9,00,8D,5A,77,A9,1,8D,5D,77,AD,5D,77,20,88,76,AD,5A,77,F0,1,60,AD,5D,77,
18,69,1,C9,6,F0,3,4C,AD,75,60
70 DATAAD,4,77,D0,3,4C,B1,74,38,E9,4,8D,4,77,85,19,C9,18,B0,3,4C,D1,74,A9,08,8D,
59,77,AE,59,77,BD,0A,77,8D,52,77,BD,B,77,8D,53,77,20,D8,77,4C,80,74
80 DATA@,7480,90,3,4C,A9,74,A9,8,18,6D,59,77,8D,59,77,C9,30,F0,3,4C,E3,75,AD,5,7
7,85,1B,A9,6,4C,14,A8,A9,0,8D,4,77,A9,6,4C,D4,A7,A9,1,8D,5E,77,4C,9F,74
999 DATA @,@
1000 PRINT"SPACE3 ｦ LOAD ｼﾃｸﾀﾞｻｲ":END
```

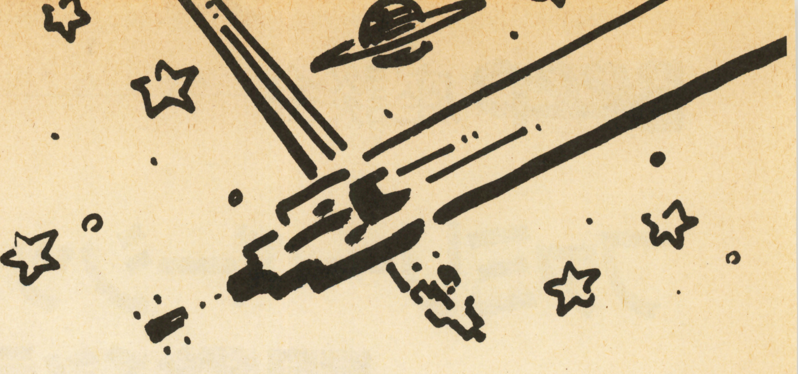
リスト3

```
10 CGSET1,1:CLEAR &H747F:A=&H74B1
20 READA$:IFA$<>"@"THENPOKEA,VAL("&H"+A$):A=A+1:GOTO20
30 CLEAR:VIEW:LOCATE5,21:PRINT"SHOOTING SPACE GAME":DATAAD,58,77,29,F0,D0,1,60,A
D,0,77,8D,4,77,AD,1,77,18,69,4,8D,5,77,A9,0,8D,5E,77,8D,5F,77,60,A9,1,8D,5F,77,4
C,9F,74,@
2000 SPRITEON:A=0:B=1:Z=8:W=&H7708:DEFSPRITEA,(A,B,A,A,A)=CHR$(164)+CHR$(165)+CH
R$(166)+CHR$(167):FORC=B TO5:DEFSPRITEC,(A,B,A,A,A)="":POKEW+Z*C,B,B,RND(100)+50
,RND(100)+50,B,B,C:NEXT:H=A:M=A
2005 POKE&H7700,200,100,A,A,A,A:POKE&H775A,A,A,A,A,A
2010 DEFSPRITE6,(A,A,A,A,A)=CHR$(223):LOCATEZ,22:PRINT"HIT :";H;:LOCATEZ,23:PRIN
T"MISS:";M;
4000 CALL&H75A6
4003 IFPEEK(&H775A)=B THEN5000
4005 IFPEEK(&H775A)=2THEN9000
4010 CALL&H7500:IFPEEK(&H775A)=B THEN5000
4020 CALL&H75CA:IF PEEK(&H775E)THEN6000
4030 IFPEEK(&H775F)THEN M=M+B:LOCATE13,23:PRINTM;:POKE&H775F,A
4090 PAUSE3:GOTO4000
5000 F=A:D=PEEK(&H7700):E=PEEK(&H7701):GOSUB8000:LOCATEZ,10:PRINT"GAME OVER":END
6000 F=PEEK(&H7759)/Z:D=PEEK(&H770A+F*Z):E=PEEK(&H770B+F*Z):GOSUB8000:SPRITEF:PO
KE&H7708+Z*F,A
6010 POKEW+F*Z,A,B,A,A,A,A:POKE&H775E,A:H=H+B:LOCATE13,22:PRINTH;:GOTO4090
8000 FOR G=A TO5:POKE&H680+4*F,180,181,182,183:SPRITEF,D,E:PLAY"T101C2":POKE&H68
0+4*F,176,177,178,179:SPRITEF,D,E:PLAY"E":NEXT:RETURN
9000 F=PEEK(&H775D):IFRND(100)>70THENPOKEW+F*Z,B,B,RND(20)+24,RND(20)+31-((F AND
1)=0)*110,RND(2)+B,RND(B)+B,RND(5)+B
9010 POKE&H775A,A:IFF<>5THENCALL&H75BC:GOTO4003
9030 GOTO4010
```

※4090行のPAUSE3の3を小さくするとゲームのスピードが速くなります。

BGグラフィック画面データ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00
1	F00																											F00
2	F00																											F00
3	F00																											F00
4	F00																											F00
5	F00																											F00
6	F00																											F00
7	F00																											F00
8	F00																											F00
9	F00																											F00
10	F00																											F00
11	F00																											F00
12	F00																											F00
13	F00																											F00
14	F00																											F00
15	F00																											F00
16	F00																											F00
17	F00																											F00
18	F00																											F00
19	F00																											F00
20	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00	F00



まう。要するに、誰でもすぐお手軽に、というわけにはいかないということである。これが機械語の欠点といえよう。

しかし、その困難を乗り越えて機械語を使いこなせるようになりさえすれば、そこには無限の可能性がひらけていく。まずスピードがベーシックの数十～数百倍。うまくプログラムできるようになると、2Kのエリアでも相当のプログラムを組むことができるようになる。

③ベーシックで使うには、

一応、ファミリーベーシックには、メモリを直接あつかったり、機械語と連絡をとったりする命令がついている。

まずPOKE文。アドレスを指定して、その内容を変更することができる。これがあれば、機械語のプログラムをメモリに書きこむことができるわけだ。POKE文には便利な機能がついていて、アドレスのあとに複数の内容を指定すると、連続したメモリの内容を変更できる。小さなデータやプログラムなら、POKE文1つですんでしまう。

次はPEEK文。アドレスを指定し、その内容を読み出す関数である。すでにメモリ内に書きこまれたプログラムやデータを読み出すことができる。

そしてCALL文。これは、機械語のプログラムのあるアドレスを指定して、そこから機械語で実行させる命令である。RTS命令(詳しいことは、機械語の本を読んで自分でチャレンジしてほしい)でベーシックへと制御がうつるようになっている。

もう1つ重要なのがCLEAR文。ベーシックで使用するメモリの上限を設定し、それ以後を機械語のプログラム領域として指定する命令である。これを実行しないでやたらとメモリの内容を変えると、コンピュータが暴走する危険があるので注意が必要だ。

以上が機械語を使うため、ベーシックに用意された命令である。

④ゲームの打ちこみ方

機械語のめんどうなところは、その打ちこみである。わけのわからない数字の列をただ打ちこんでいくのは、かなりの苦痛だ。しかも、1ヵ所でも間違えると、エラーのメッセージも出ないまま、あっさり暴走ということにもなりかねない。打ちこみには、ベーシックのプログラム以上の確認が必要だ。

機械語のプログラムを入力する方法をいろいろためしてみたが、セーブ、ロードのしやすさなどを考えて、ベーシックのデータ文で機械語のプログラムを入力し、次にそれをメモリ上に置くプログラムを入力し、実行する。そして、NEW↵としてから、こんどは、ゲーム本体を動かすベーシック部分のプログラムを入力して、実行するという方法をとることにした。

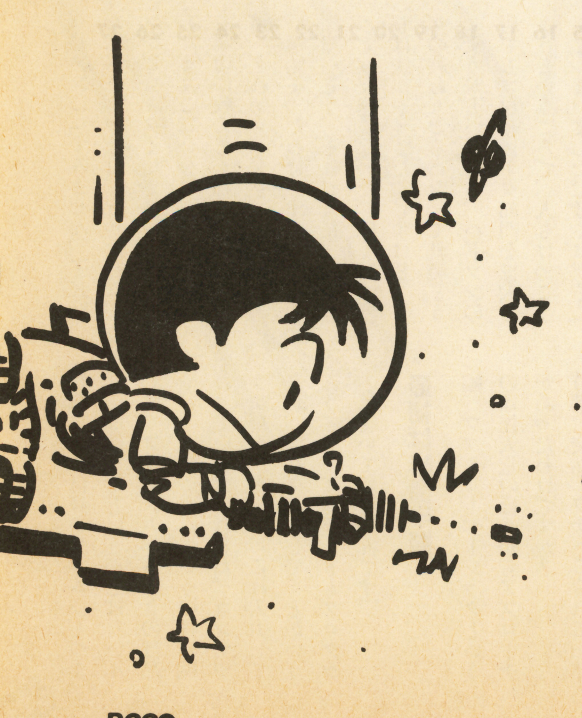
プログラムリスト1、2、3をそれぞれ入力して、セーブしてほしい。まずプログラム1をロードしてRUN。次にプログラム2をロードしてRUN。この2本で機械語プログラムを作るわけだ。プログラム3をロードして、BGを入力し、ゲームスタートだ。

⑤ゲームの遊び方

機械語ならではの高速性を生かしたゲームを作りたかったのだが、ファミリーコンピュータの画面コントロールがまだ完全に理解できていなかったため、一部ベーシックにたよるハメになってしまった。ベーシックの解析がさらに進めば、すべて機械語のゲームも夢ではないのだが…。

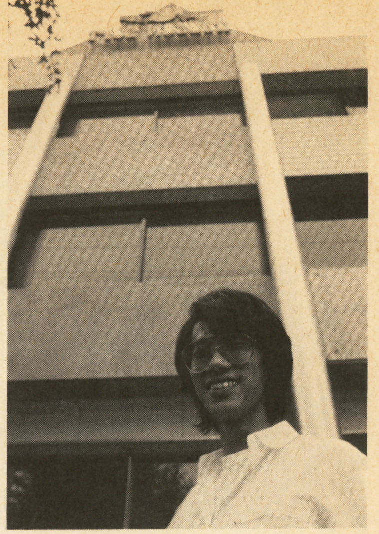
☆ストーリー☆

ここは、宇宙軍戦闘部隊の無重力戦シミュレーションルーム。キミは、スターシップに乗ってただよっている。と、急に四方からスピナーの攻撃が………。むかってくるスピナーを次々とかわし、レーザーで破壊しよう。ただし、スターシップの向きは変えることができず、レーザーは前方のみに発射できる。そして運動は、加速運動になっているから気をつけよう。✚ボタンで移動したい方向を指定する。止まるときには、運動方向と逆の方向を指定する。ずっと押しっぱなしだとどんどん加速してしまうので気をつけよう。レーザーはAかBのボタンで発射だ。



Beep 座談会

ファミリーベーシック開発秘話



ゲスト㈱ハドソン・竹部隆司さん

きききて：浮田利康／KAZ／BEEP編集室

編 きょうは、ファミリーベーシックを、任天堂、シャープと共同開発したハドソンの竹部さんに来ていただきました。ファミリーベーシック開発の裏話や、ユーザーへのアドバイスなど、ざっくばらんにお話していただきます。

ききてには、BEEPの人気ライターKAZさんや浮田さんにも加わってもらいます。

…ちょっと難しいところもあるけれど、がんばって読もうネ。

ハドソンのはじまり

編 むかしの話になりますが、竹部さんは、ハドソンの創設のときから、いたわけですか。

竹部 ハドソンはもともとアマチュア無線の販売店を北海道でやっていたんだよね。無線一品で商売してた。社長がむかし、コンピュータにはけっこう関係あったんです。アメリカにわざわざ出かけてマシン買ってきたり。そして、これからはこれだ！って思ったらしい。ぼくはそのころまだ学生で、友達が「最近マイクロ・コンピュータってのがあるよ」って教えてくれておもしろそうだなと感じて、TK80を買おうと思ったけど、どこで売ってるのかわかんない。ま、無線屋に行けば、コンピュータも売っているだろうと安易な考えで、ハドソンに「すみません、TK80ください」ってね、行ったわけ。コンピュータくださいなんて客はあんまりこないわけで(笑)、とても親切に教えてくれました。

編 それからハドソンとのつきあいが始まったわけですね。

竹部 そう、それで社長と話があってお友達になった。ぼくは大学では法律やってたんですけど、3年の終わりごろになって就職どうしようかというころ、ハドソンの店でもコンピュータ部門を拡大しようというところで、人もけっこう集まっていました。ぼくは、そこでシャープのMZ80で遊んだりゲームなんかを作っていて、マシンを買いにきた客に、「こんなので遊べますからやってみてください」ってサンプルなんかを貸してあげてたんです。そのうち、これ売れるじゃない、という話になってきて、新しいソフトの注文がきたり……ちょうどそのころ大学も4年になっていたんで、そのまま……というわけです。

そのころは今ほど大きくなるなんて思っていなかったですけどね。そのうち、北海道から東京に営業所ができてこちらに移ったりといったぐあいですね。

編 関わってこられたゲームはどんなものですか？

竹部 ぼく自身がゲームひとつまるまる作ったということはないんです。プログラム作るときの部品やパッケージなんていうのには関わっていますけれどね。コンテやラフなんか書いて、これ入力してとか、プログラムして、とかいうふうで、デゼニランドもプログラムは書いていないんです。

最初は半信半疑

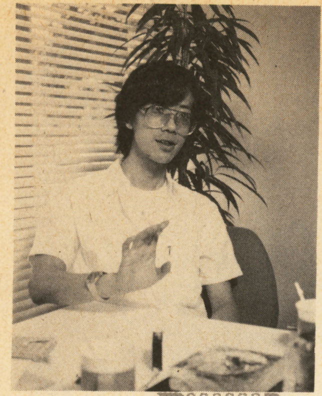
編 ファミリーベーシックというのは任天堂のほうからアプローチがあったわけですか？

竹部 突然、ウチへこられて一緒にベーシックを作ってくださいというわけです。当時ファミコンは出たばかりで、カセットも、ドンキーコングとそのジュニア、それにポパイの3本しかなかった。おもしろいところからでるんだナということで、仕様なんかの説明きいても、最初は半信半疑で、いけるかなあと思いましたね。でもまあマシン的にはとてもおもしろいもんだという感じはありました。

編 それは83年の8月ごろですか？　システムとしてはハドソンがぜんぶやられたんですか？

竹部 任天堂さんとシャープさんとハドソンの共同開発、ということで。

大まかな案が任天堂からきて、



こういう命令は入るとか入らないとか、仕様を作ったわけです。任天堂さんは、ファミリーコンピュータはゲームマシンであると、あくまでもゲームをやるマシンで、本格的コンピュータへの入門になっていればいいということですね。ですから思いきって命令切っちゃったとこなんかもありますしね。

編　大ざっぱなコンセプトだけは任天堂のほうからきたわけですね。

竹部　ええ。それでプログラムの勉強もできるんだヨ、というふうにぼくは了解していました。そういったコンセプトで作ったベーシックですから、まあ普通の基準から考えれば変態的な(笑い)ベーシックですよね。

KAZ　でも動きがあるし、ほかのベーシックは機能をもたせすぎているところもあるし……。

竹部　みんなもMSXは意識してたと思うんですよ。ファミコンの倍以上はしますよね。しかもあちらはレッキとした電気メーカーのものだと。こちらはオモチャなどを作っているメーカーが作るわけだから、それなりの特色を出さなくちゃと思いました。

1年ぐらいかかったかな

編　ところでベーシックを具体的に作るときの方針は?

竹部　まず、ユーザーがかんたんにキャラクターを動かせるようにする、ということですね。それであのMOVEという命令を作った。MSXで、あれと同じことをやろうとすると、あの千倍のプログラムが必要だよ。ベーシックだとダイレクト命令でもキャラクターを動かすことができるんだよ。そのへんが考え方の相違でね。

編　依頼をうけてどのくらいの期間で作ったんですか。

竹部　最初の話からはなんだかんだ1年ぐらいかかったかな。コードを書いてる期間は10ヵ月ぐらいかなあ。その前にはバグってというのが実情で。ベーシック自身は難しい構造してないからできるんだけど、バグるのね。

開発の苦労

編　ラムがすくないというのが一番たいへんだった?

竹部　そう、X1とかPCあたりでベーシック作っているんだったら、通常考えなくてもいいことを、ここでは悩まなくっちゃいけない、いろいろの制約があるんですよ。ゲームを作るには便利なことなんだけど、プログラムがふくれるわけです。ロムなもんなのでピタ・・・1バイトまからん。ちょっと書くとプッとはみだす、バグつぶしてアセンブルしてみるとプッとはみだしてる。バグるわ、はみだすわのメモリとの戦い、つぎは納期との戦い、時間は迫るわ、バグってるわ、はみだしてるわ。「すんません、もう1日……」とかいって……。

逆にボクのほうから、実際にこのベーシックを使ってプログラムやってる人に尋ねたいんだけれど、どうですか?

KAZ　2キロしかないというミニサイズのわりには、けっこう動くものが作れるのでびっくりした。打ちこんでみるとつめこみがきく、けっこう入る。プログラムを解析してみてわかったんですけれど、わかりやすいんですね。

竹部　8Kはぜったい必要だって主張したんです。そうすれば、スタートレックができるでしょ。あれはタイニーベーシックの基準点だと思っているから。

KAZ　ファミリーベーシックでスタートレックをしたかったというわけ?

竹部　したかった!!

浮田　PEEK文とかPOKE文とか用意されていますけれど、マシン語の利用とかは多少意識されていたんですか?

竹部　しゃれてです!!　CALL文は全部で9バイト、なにかやりたいときには、ツールをうめこんでコールできれば便利だナという感じで。ユーザーにとってはシャレだなと思って作ったの。

浮田　ほとんどメモリ不足ということですけど。MOVE文がスプライト8枚分で決まっているでしょ、スプライト文でMOVE文も少し使えるような形になれば、これはすごいと思うんだけど。

竹部　それは考えたね。でも実験してみたらすごいメモリくってダメだったんだ。

今ふりかえって

編　作っていた当時と今との、技術的な落差みたいなものがありますか?　たとえば、今ふりかえって、こんなふうにしておけばよかったなあーとか…

竹部　最初にもらったファミリーベーシックの資料はごくかんたんなものだったから、自分でマニュアルに従って解析して理解していった。今考えるともうちょっとうまくプログラム作れるなあと。だんだんマシンのアナというか、わかってくるんですよ。技術の向上のためにはマシンと深く関わって遊ばなくっちゃいけないのね。当時は目いっぱいやったつもりなんだけど、今になるともっとおもしろい機能を発見したり、もっとサイズが短くていいもの作れる。

あきらめるな

編　ファミリーベーシックのユーザーというのは、おもに小学校高学年から中学生だと思いますけど、彼らに対して開発した立場からなにかアドバイスを。

竹部　作っていてあきらめないことだね。行きづまっても、必ず別な方法があるから、別な発想をしてみると必ず攻略法があるんだ。このベーシックをやってるときだってやっぱりそうなんだ。メモリがないときは、スペースを切ったり、マルチステーティングしてみる、ムダな命令はないかナと考える。それでダメだったらつぎはあるひとつのブロックをとり出してきて、一番サイズの短くなるようなアルゴリズムの組み替えをやってみるんだ。

編　それは初級者にもできますかネ。若き諸君たちにも。

竹部　うん、だからやっぱりあきらめないでネ。

　必ず短くできる、自分の考え方を整理してみて、やりたいことを分析して、やりたいことを実現するための部品の品定めをして、構築したアルゴリズムがうまくいかなければ、その部品を組み替えてみるということだね。

　ハドソンのスタッフはそれぞれプログラム書いているんだけど、やってみると副産物というのができるんですよ。それは、当人にはそのとき不要でも別な人間には必要な部分のこともあるのね。「ちょうだい」ってもらいにいったりね。あいつの、このあいだのあの部分が使えるナと思うと、「ちょうだい」ってもらうわけ。あとはマシンは88だからパッケージちょうだいねってキャラクター入れて、あと接着剤をつけるようなモンです。

編　ファミリーベーシックの今後についてどうですか?

竹部　ぼくの子供(つまりファミリーベーシックのこと)がこれだけ世の中にでているってったらコワイですね。ロム出しする前の晩ていうのは眠れなかったですよ。

編　バージョン1があってすぐバージョン2になったでしょ。短期間で……。

竹部　あれはバグったんでなくて、この命令を急きょ入れようということでね。バージョン2が終わって、やっと済んだなあと思ったらすぐバージョン3で、今度はメモリが倍になりますって。

編　なにか言い残したことありますか?

竹部　みんなが、ほんとにプログラムっておもしろいナって感じたらいいなと。プログラムのおもしろさをわかってほしい。その上でMSXでも他の上級のコンピュータにでも移ってもらえればいいと思っているんです。プログラムするっていうのは、ロジカルな思考をするということですから。

大好きソフト

編　それでは最後にファミコンゲームでどれが一番好きですか、ベスト3をどうぞ。

竹部　最近のはちょっとおもしろくないみたいで。スーパーマリオはおもしろいネ。ゴルフもいいネ。テニスはおもしろかった。あとはスターフォースかな。

　ファミコンゲーム最近の見ると、悩んじゃうよ。今は過渡期、ゲームの質がやや落ちてるし、乱作ぎみかナ。この時期をぬければ、きっとすごいソフトがでてくるよ。

KAZ　ボクのベスト3は決まりました。F1レース、バルーンファイト、テニスかな。

浮田　条件つきでドルアーガ。あとスーパーマリオとゼビウスかな。

竹部　でもゼビウスが32Kでスターフォース16Kなんだよね。スターフォースはやっぱりあの圧縮に評価を与えるね。つめこみ技術はゼビウス以上だよね。あれは32Kでやればどうってことないんだよね。あれは中をみるとすごいの、グチャグチャつめこんであってほとんどサッパリわかんないんですよ。データなんか二重三重の圧縮に加えて、タイプ別の圧縮とかもやってる。よくあれだけ入ったよ。

編　買ったけどベーシックを使わずに寝かしているユーザーに呼びかけてください。

竹部　BEEPを読みなさい!

　コンピュータというのは受動的な機械じゃないのね。勉強してよいプログラムを書けばそれだけの動きをするし、そうしなければ動かない。

編　でもね、そういうマシンから脱皮しないといけないんじゃない?

竹部　ベーシックも情報工学というひとつの学問体系からなりたっているわけ。ちょっとかじったぐらいじゃわかりっこないというのが本音。でも使っていると動く。そして、動かしているうちに知らず知らず覚えていく。それを大事にしたいね。ベーシックというのも捨てたもんじゃない。

KAZ　制約が多くて、パズルとして遊ぶにはおもしろい。

竹部　マゾの世界というのはほんとう、逆境の中で遊びを作りだしていくというところかな!

ファミリー ベーシック
おもしろゲーム大作戦

定価2,600円
1985年12月10日初版発行

浮田利康

生まれは東京。育ちも99%東京という、99%東京人。都内の某大学の大学院に通う学生だったりする。ハードのハの字も知らないソフト専門屋なのに、何故かハードをやっていたとか思われ、困っている今日このごろです。パソコン歴5年、ファミリーベーシック歴1.5年。

KAZ

石川県は、金沢市、犀川の水で産湯を使い、以来したいほうだい不自由なくここまで生き抜いて来た20代半ばの平均人間、それがKAZである。ファミリーベーシックにとりつかれて1年あまり、このまま精気を吸いとられていくのでしょうか？

●本書の内容についてのお問い合わせは、往復ハガキか封書(返信用封筒同封)で、下記へお願いします。

〒102 東京都千代田区四番町2-1
㈱日本ソフトバンク 出版部
「ファミリーベーシック
おもしろゲーム大作戦」係

※ファミリーベーシックは任天堂の商標です。

表紙イラスト●下田一貴
表紙デザイン●勝俣正希
ビデオアート●山本実

編集●BEEP(豊田素行／麻生健司／白勢房枝／新保利昭／武田浩二)／アートサプライ
レイアウト●アートサプライ
イラスト●国本雅之／青木美和
カメラ●杉山和美／斉藤ふみお
校正●関民子＋グループごじら

著者●浮田利康／KAZ
発行者●孫 正義
発行所●株式会社日本ソフトバンク
出版部 〒102 東京都千代田区四番町2-1 ☎03-261-4095
印刷所●(株)東京博文堂

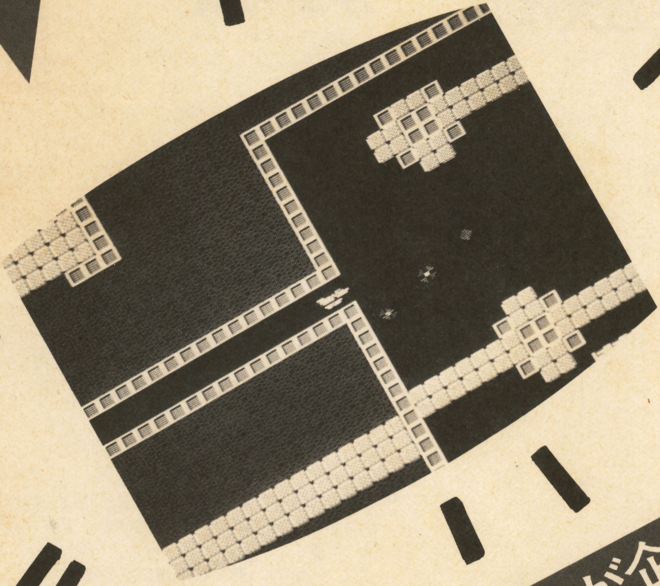
全国ファミコンファンの皆さん
おまちどーさま！

別冊Beep
ファミリーコンピュータ™ '86ゲームハンドブック

BEEPが送る1冊まるごとファミコンブック

人気ゲームのテクニックやマップがたくさんあるぞ

キミのスコアやマップも書きこめるベンリな1冊



カラテカ／キン肉マン／テグザー／ガル／ポートピア連続殺人事件／ルナーボール／グラディウス／ツインビーほか人気ゲームをくわしく紹介

Beepが企画するファミコンの本

好評発売中！



SOFT BANK 日本ソフトバンク☎03・261・4095

日本ソフトバンクの雑誌

月刊(8日発売) Oh!16



パソコンの本格的活用のための16ビットのハードおよびソフトの活用をわかりやすく紹介し解説。

月刊(18日発売) Oh!FM



人気No.1のFM-7をはじめ新機種FM-NEW7、77やFM-8、11などFMシリーズ専用の情報誌。

月刊(18日発売) Oh!MZ



人とコンピュータ、ソフトとハードを結ぶ、シャープ・パーソナルコンピュータの専門誌

月刊(18日発売) Oh!PC



これからはいよいよソフトウェア充実の時代。Oh!PCは時代にマッチしたソフトウェアのかたまり。

季刊 Oh!HIT BIT



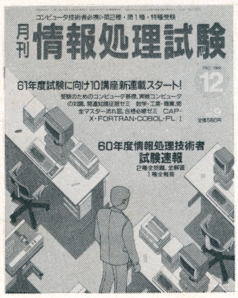
コンピュータを中心としたパーソナル・レベルでのコミュニケーションを追求したソニーパソコン専門誌。

季刊 Oh!HC



ホットな情報を満載したEPSON専門情報誌。あらゆる角度からパソコンの世界をみつめています。

月刊(8日発売) 月刊 情報処理試験



情報処理技術者第2種・第1種・特種試験の受験を志す人にうってつけの親切雑誌。

月刊(8日発売) Beep



パソコンを生き返らせ、日本全国をゲーム症候群に巻き込む。BEEPはパソコンの活性剤です。

ご注文の方法

雑誌・書籍を問わず、ご注文は最寄りの書店でお願いいたします。小社に直接お申し込みいただく場合は、定価代金と郵送料を現金書留でお送りください。年間購読のお申し込みは、各誌綴じ込みの振替用紙をご利用ください。なお、ご不明な点はお問い合わせください。

TEL03(261)4095(営業)

●Oh!PC別冊 Selection Oh!88

Oh!PCに掲載されたPC-8801シリーズの記事の中からベストセレクトしたオール88の決定版。

●Oh!PC別冊 Selection Oh!98

Oh!PCに掲載されたPC-9801シリーズの記事の中から、とくに人気のあったものだけを収載。

不定期刊 Oh!PASOPIA



パソピアシリーズを使用しているあなたにとって、「Oh!PASOPIA」は魔法の箱です。

SOFT BANK

おもしろゲーム大作戦　テープリスト

ゲーム名	BG画面データのファイル名	プログラムファイル名（ゲームファイル名）
ファミリーベーシックタイプ特訓ゲーム	TYPE␣BG	TYPE
ガンバレお姉さん!! 炎のビープ番	BEEPBAN␣BG	BEEPBAN
気分はもうデバッガー・バグシューター	BUG␣BG	BUGSHOOT
なみだのジューススタンド	JUICE␣BG	JUICE
マリオとルイージのVSバスケット	BASKET␣BG	BASKET
21(ファミコン風ブラックジャック)	BJBG	BJ
ルイージの三段とび	3ダントビBG	3ダントビ
一発勝負! ハイ&ロー	HIGHBG	HIGH
マリオの走り幅とび	BG␣HABATOBI	HABATOBI
カジノもまっ青? スロットマシン	SLOTBG	SLOT
レディの走り高とび	HIGHJUMPBG	HIGHJUMP
シューティングゲーム		SHOOTING
数字をあてろ!	MASTER-MINDBA	MASTER-MIND
ファミリーおもちゃ箱	オモチャBG	オモチャ
ペンジンのペンギンズバー		PENGUIN
色出し活用パレットツール(ツール)		PALETTOOL
かっとびモトクロッサー	MOTO␣BG	MOTO
VS2次元ボクシング	BOXING␣BG	BOXING
むかしなつかし VSテレビピンポン	PING␣BG	PINGPONG
高速シューティング・スペースゲーム	SPACE␣BG	SPACE1・SPACE2・SPACE3　（計3本）

●ゲームをロードするときは、このファイル名を入力してください。
くわしいロード方法は，9ページを読んでください。
●ファイル名にある␣はスペースを表します。
●ロード中にエラーがでたときは、音量を変えてロードしなおしてください（B面にも同じプログラムが入っていますので、B面でもためしてみてください）。

SOFT BANK



A
ファミリー ベーシック™
おもしろゲーム大作戦
（B面にも同じ内容のプログラムが入っています）
ファミリーベーシックは任天堂の商標です。
Beep
Ⓒ㈱日本ソフトバンク 不許複製

「マリオの走り幅とび」「ボクシング」「ハイ&ロー」「ペンギンズバー」など人気ゲーム19本入

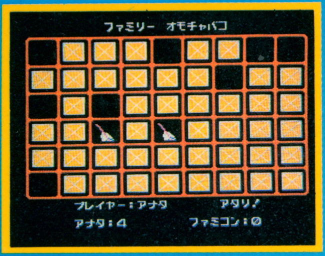
Beepに載ったプログラムのほか、新作ゲームも入ってる!(カセットテープ1本+本88ページ)●バージョン2用

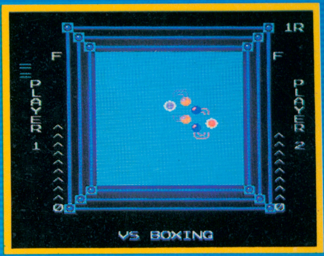
楽しいファミリーベーシック講座(全20章)機械語入門付

Beepの人気ライター・KAZ兄ィや浮やんが、ファミリーベーシックのゲーム・テクニックをわかりやすく紹介!

特別座談会/ユーザーへのアドバイスや開発のこぼれ話

ファミリーベーシックの開発にたずさわったハドソン・竹部さんが大いに語る。





初級編●ゲーム名

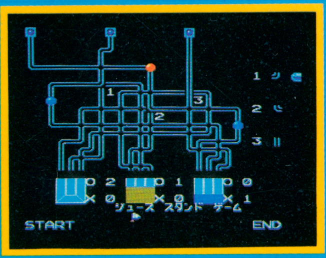
- ファミリーベーシックタイプ特訓ゲーム
- ガンバレお姉さん!! 炎のビープ番
- 気分はもうデバッガー・バグシューター
- なみだのジューススタンド
- マリオとルイージのVSバスケット

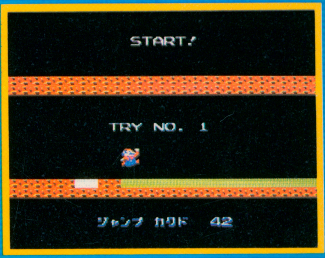
中級編●ゲーム名

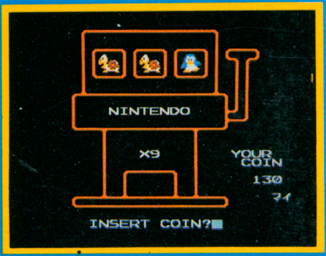
- 21(ファミコン風ブラックジャック)
- ルイージの三段とび
- 一発勝負! ハイ&ロー
- マリオの走り幅とび
- カジノもまっ青? スロットマシン
- レディの走り高とび
- シューティングゲーム
- 数字をあてろ!
- ファミリーおもちゃ箱

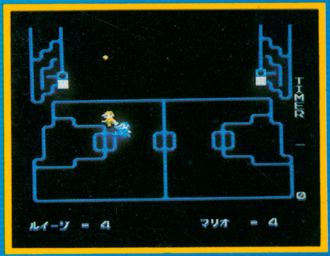
上級編●ゲーム名

- ペンジンのペンギンズバー
- 色出し活用パレットツール(ツール)
- かっとびモトクロッサー
- VS2次元ボクシング
- むかしなつかしVSテレビピンポン
- 高速シューティング・スペースゲーム









ファミリーベーシックおもしろゲーム大作戦/Beep編集 定価2600円

日本ソフトバンク SOFT BANK

ファミリーベーシック™

Beep

おもしろゲーム大作戦

SOFT BANK

